Determination of Inner and Outer Bounds of Reachable Sets

Francisco Fernandes Castro Rego

Dissertação Para a obtenção do Grau de Mestre em Engenharia Aeroespacial

Júri
Presidente: Prof. João Manuel Lage de Miranda Lemos
Orientador: Prof. António Manuel Santos Pascoal
Vogal: Prof. José Raul Carreira Azinheira

Maio de 2011
Abstract
To verify the safety of a dynamic system, one important procedure is to compute its reachable set of states. There are different methods of computing reachable sets, namely interval integration, capture basin, methods involving the minimum time to reach function and level set methods. This work deals with interval integration to compute subpavings to over or under approximate reachable sets.

An algorithm to over and under estimate sets through subpavings, which potentially reduces the computational load when the test function or the contractor is computationally heavy, is implemented and tested. This algorithm is used to compute inner and outer approximations of reachable sets. The test function and the contractors used in this work to obtain the subpavings involve guaranteed integration, provided either by the Euler method or by VSPODE.

The devised methods were applied to compute inner and outer approximations of reachable sets for the double integrator example. From the results it was observed that using contractors instead of test functions yields much tighter results. It was also confirmed that for a given minimum box size there is an optimum time step such that with a greater or smaller time step we obtain worse results. Unexpectedly, when using VSPODE for the guaranteed integration we obtain worse results than with the Euler method, suggesting a possible flaw in the application of the program or in the program itself.

Keywords: Reachable sets, Interval Analysis, Subpavings, Guaranteed integration

Resumo
Para verificar a segurança de um sistema dinâmico, um procedimento importante é o calculo do seu conjunto de atingibilidade. Existem diferentes métodos de calcular conjuntos de atingibilidade, nomeadamente os designados como “interval integration”, “capture basin”, “level set” e métodos que envolvem a função de tempo mínimo de alcance.

É implementado e testado um algoritmo para sub e sobrestimar conjuntos, que potencialmente reduz o esforço computacional quando a função de teste ou o “contractor” é pesado computationalmente. Este algoritmo é usado para calcular limites exteriores e interiores de conjuntos de atingibilidade. A função de teste ou os “contractors” usados neste trabalho incluem “guaranteed integration” que é realizada através do método de Euler ou pelo VSPODE.

Os métodos desenvolvidos foram aplicados para calcular aproximações internas e externas de conjuntos de atingibilidade para o exemplo do duplo integrador. Através dos resultados obtidos foi possível concluir que o uso de “contractors” em vez de funções de teste conduziu a resultados muito mais ajustados e que para um dado tamanho mínimo de caixa existe um passo temporal óptimo. Ao contrário do esperado, o método de Euler para a “guaranteed integration” revelou melhores resultados que o VSPODE.

Palavras-chave: Conjuntos de atingibilidade, Interval Analysis, Subpavings, Guaranteed integration
Acknowledgments

I would like to thank my supervisors dr. Elwin de Weerdt and dr. Eddy van Oort for introducing me to the fields of reachability and interval analysis and for their patience and contribution for this work. I would like to thank dr. Ir. Erik-Jan van Kampen for his assistance in wrapping up this thesis. I would also like to acknowledge the contribution of dr. Amilcar Miranda for his review of the first draft of the thesis and for his words of encouragement. A special thanks to dr. António Pascoal for introducing me to the field of control and for his acceptance to be the co-supervisor at IST. A word of recognition to dr. Qi-Ping Chu for his trust in considering me for addressing the subject of this thesis.

Finally I would like to express my gratitude to my family and friends for their continued support.
## Contents

Abstract .................................................................................................................. II
Resumo ................................................................................................................... II
Acknowledgments .................................................................................................. III

List of Figures ......................................................................................................... VI
List of Tables ......................................................................................................... VIII
Nomenclature ......................................................................................................... IX

Chapter 1: Introduction ......................................................................................... 1
  1.1 Background .................................................................................................... 1
  1.2 Problem Definition ...................................................................................... 2
  1.3 Thesis goal ................................................................................................... 3
  1.4 Thesis Outline .............................................................................................. 3

Chapter 2: The Problem - Reachable Sets ............................................................ 5
  2.1 Definition ..................................................................................................... 5
  2.2 Hamilton-Jacobi-Isaacs (HJI) PDE ............................................................... 7
  2.3 Alternative formulations .............................................................................. 8
  2.4 Available methods ...................................................................................... 9
    Level set methods ............................................................................................ 9
    Neural approximation of PDE ........................................................................ 11
    SOS feasibility ............................................................................................... 13
  2.5 Conclusions ................................................................................................ 14

Chapter 3: The Approach - Interval Analysis ....................................................... 14
  3.1 Introduction to Interval Analysis .................................................................. 15
  3.2 Inclusion functions ...................................................................................... 18
  3.3 Guaranteed Integration ............................................................................... 20
  3.4 Interval Taylor Models ............................................................................... 22
  3.5 Available Solvers ...................................................................................... 24
  3.6 Subpavings ................................................................................................ 26
  3.7 Conclusions ................................................................................................ 29

Chapter 4: The Proposed Solution ................................................................. 30
4.1 Theoretical background - Method of Jaulin .............................................................. 30
4.2 Formulation for Reachable sets ................................................................................. 32
4.3 Conclusions .................................................................................................................. 38

Chapter 5: Computational Application ............................................................................. 38
5.1 Branch and prune (B&P) algorithm ............................................................................. 38
    Operational Results ......................................................................................................... 43
5.2 Branch and prune (B&P) algorithm with flag ............................................................. 48
    Operational Results ......................................................................................................... 51
5.3 Improving Computational Efficiency ......................................................................... 54
    Operational Results ......................................................................................................... 56
5.4 Branch and Contract (B&C) ....................................................................................... 58
    Operational Results ......................................................................................................... 62
5.5 Outer bound .................................................................................................................. 65
5.6 Inner bound ................................................................................................................... 67
5.7 Applying the Euler Method ......................................................................................... 68
5.8 Applying Interval Taylor Models ................................................................................ 69
5.9 Conclusions .................................................................................................................. 70

Chapter 6: Application Example ....................................................................................... 71
6.1 Analytical Solution ......................................................................................................... 71
6.2 Branch and Prune ......................................................................................................... 73
6.3 Branch and Contract .................................................................................................... 76
6.4 VSPODE ....................................................................................................................... 78
6.5 Conclusions .................................................................................................................. 79

Chapter 7: Conclusions and recommendations ................................................................ 81
7.1 Conclusions .................................................................................................................. 81
7.2 Recommendations ........................................................................................................ 82

Bibliography ....................................................................................................................... 83

Appendices .......................................................................................................................... 85
Appendix A: mean value evaluation ................................................................................... 85
Appendix B: Tables with data from algorithms ................................................................. 86
List of Figures
Figure 2-1: Backward (left) and Forward (right) reachable sets .............................................. 6
Figure 2-2: Illustration of the definition of Safe Flight Envelope (Safe Maneuver Set) .................... 8
Figure 2-3: Results of the pursuer-evader system reachable set from different angles .................. 10
Figure 2-4: Results of the pursuer-evader system reachable set from different angles .................. 11
Figure 2-5: Training points around the boundary of the initial set (rectangle) .............................. 12
Figure 2-6: Result from the neural approximation of the HJI PDE ........................................... 12
Figure 3-1: Example of the dependency problem ......................................................................... 17
Figure 3-2: Example of the wrapping effect .................................................................................. 17
Figure 3-3: Example of mean value form ..................................................................................... 19
Figure 3-4: Example of slope form ................................................................................................ 19
Figure 3-5: Interval bounding of a function by a Taylor model: orders 7 (left) and 10 (right) ........ 24
Figure 3-6: Conventional interval bounding of a function: from top left to bottom right the number
of interval evaluations is 100, 400, 1600 and 6400 ................................................................. 24
Figure 3-7: Binary tree ................................................................................................................... 26
Figure 3-8: Subpaving of x0 .......................................................................................................... 27
Figure 4-1: Figure illustrating the principle of \[ q \] ........................................................................... 31
Figure 4-2: Illustration of theorem V ............................................................................................ 31
Figure 4-3: Result of the algorithm of method of Jaulin on the Zermelo's problem ..................... 32
Figure 4-4: Illustration of assumption II ...................................................................................... 34
Figure 4-5: Illustration of assumption III ...................................................................................... 35
Figure 5-1: Flowchart of the branch-and-prune algorithm .............................................................. 40
Figure 5-2: Evolution of lists \[ S \] and \[ E \] .................................................................................... 40
Figure 5-3: Binary tree representing lists \[ S \] (solid line) and \[ E \] (dashed line) .............................. 41
Figure 5-4: Result of the example using \[ \varepsilon = 0.6 \] ................................................................. 41
Figure 5-5: Subpaving \[ S \] obtained with boxes on \[ J \] not in contact with \[ T \] ................................. 42
Figure 5-6: Subpaving \[ S \] obtained with some boxes on \[ J \] in contact with \[ T \] ......................... 42
Figure 5-7: Subpaving \[ S \] obtained without any box of \[ J \] in contact with \[ M \] ....................... 43
Figure 5-8: Subpaving \[ S \] obtained with one box of \[ J \] in contact with \[ M \] ............................ 43
Figure 5-9: Time to run (seconds) as a function of the minimum box size and number of
dimensions ................................................................................................................................. 44
Figure 5-10: Number of boxes on the result as a function of the minimum box size and number of
dimensions .................................................................................................................................. 45
Figure 5-11: Processed boxes during the algorithm as a function of the minimum box size and
number of dimensions ................................................................................................................ 45
Figure 5-12: time to run as a function of the square of processed boxes ....................................... 46
Figure 5-13: Coefficients as a function of the number of dimensions ......................................... 47
Figure 5-14: Predicted and observed values of \[ \log(tr) \] as a function of mbs ............................... 47
Figure 5-15: Predicted and observed values of \[ \log(nb) \] as a function of mbs ............................. 48
Figure 5-16: Predicted and observed values of \[ \log(pb) \] as a function of mbs ............................. 48

VI
Figure 5-17: Fluxogram of the Branch and Prune algorithm with flag.............................. 50
Figure 5-18: Overestimation of time independent capture basin using B&P algorithm .......... 50
Figure 5-19: Overestimation of time independent capture basin using B&P algorithm with flag 51
Figure 5-20: Relation between processed boxes and time to run for 1 dimension .................. 52
Figure 5-21: Comparison of time to run between B&P (blue) and B&P with flag (green) ....... 52
Figure 5-22: Comparison of processed boxes between B&P (blue) and B&P with flag (green). 53
Figure 5-23: Time to run for the modified algorithm (left) and number of processed boxes (right) ............................................................................................................. 53
Figure 5-24: Coefficients of models for B&P with flag as a function of the number of dimensions ............................................................................................................................. 54
Figure 5-25: Time to run for the improved algorithm............................................................ 56
Figure 5-26: Time to run of B&P algorithm before alterations (blue) and after alterations (green) ............................................................................................................. 57
Figure 5-27: Time to run vs. number of processed boxes for one dimension ....................... 57
Figure 5-28: Logarithm of time to run vs. logarithm of processed boxes for dimensions greater than one.................................................................................................................. 57
Figure 5-29: Flowchart of the branch-and-contract algorithm .......................................... 59
Figure 5-30: Evolution of list $\mathcal{S}$ ....................................................................................... 60
Figure 5-31: Final outer subpaving $\sigma$ ................................................................................. 60
Figure 5-32: Binary tree representing list $\mathcal{S}$ .................................................................. 60
Figure 5-33: Subpaving $\mathcal{S}$ obtained with $\text{rin2} = 0.25$ ...................................................... 62
Figure 5-34: Subpaving $\mathcal{S}$ obtained with $\text{rin2} = 0.5041$ ..................................................... 62
Figure 5-35: Time to run for one dimension vs. processed boxes ........................................ 63
Figure 5-36: Logarithm of time to run vs. logarithm of processed boxes .............................. 63
Figure 5-37: Time to run (seconds) as a function of the minimum box size and number of dimensions .................................................................................................................. 64
Figure 5-38: Number of boxes on the result as a function of the minimum box size and number of dimensions ........................................................................................................... 64
Figure 5-39: Processed boxes during the algorithm as a function of the minimum box size and number of dimension ........................................................................................................... 64
Figure 5-40: Coefficients of models for B&C as a function of the number of dimensions ...... 65
Figure 5-41: VSPODE guaranteed integration results ............................................................... 70
Figure 6-1: Analytical solution for target set (0,0) ................................................................. 72
Figure 6-2: Analytical solution for target set $([-0.2,0.2],[-0.2,0.2])$ .................................... 73
Figure 6-3: Over-approximation of reachable set computed with one time step ................. 74
Figure 6-4: Over-approximation of reachable set computed with 10 time steps .................. 74
Figure 6-5: Under-approximation of reachable set computed with 10 time steps ............... 75
Figure 6-6: Inner and outer bound of reachable set for the double integrator example using the B&P algorithm ........................................................................................................... 76
Figure 6-7: Double integrator example with one time step using contractors .................... 76
Figure 6-8: Double integrator example with 10 time steps using contractors .................... 77
Figure 6-9: Double integrator example with 20 time steps using contractors .................... 78
Figure 6-10: Double integrator example using VSPODE ............................................. 79

List of Tables
Table 5-1: List $E$ after branch-and-prune algorithm .................................................. 41
Table 5-2: List $S$ after branch-and-prune algorithm .................................................. 41
Table 5-3: Coefficients for the model on the form $a + b \cdot \log(mbs)$ ................................. 46
Table 5-4: Coefficients of models for Branch and Prune algorithm with flag .................... 53
Table 5-5: List $E$ after branch-and-contact algorithm ................................................ 61
Table 5-6: List $S$ after branch-and-contact algorithm ................................................ 61
Table 5-7: Coefficients of models for Branch and Contract algorithm ............................ 65
Table 6-1: Overestimation and Underestimation of different methods ............................... 80
Table 6-2: Overestimation and Underestimation of different numbers of time steps .......... 80
## Nomenclature

- $x(\cdot)$: State of the dynamical system
- $u(\cdot)$: Control input
- $d(\cdot)$: Disturbance input
- $S_0$: Initial set
- $T_0$: Target set
- $U$: Set of admissible control inputs
- $U$: Bound on control input
- $D$: Set of admissible disturbance inputs
- $D$: Bound of disturbance input
- $\Sigma$: Set of samples of disturbance input
- $t_f$: Final time
- $\gamma$: Control strategy
- $S(\cdot)$: Forward reachable set
- $\theta(\cdot)$: Picard iterate
- $\varphi_f(\cdot, \cdot)$: Set function of forward reachable set
- $\varphi_b(\cdot, \cdot)$: Set function of backward reachable set
- $\sigma$: Safe flight envelope
- Capt$_f(\cdot, \cdot)$: Capture basin
- $T(\cdot)$: Minimum time to reach function
- $t(\cdot; \cdot, \cdot)$: Time to reach function
- $\varphi(\cdot; \cdot, \cdot)$: Forward trajectory
- $TD(\cdot)$: Discounted minimum time to reach function
- $H(\cdot, \cdot)$: Hamiltonian
- $w(\cdot)$: Interval width
- $m(\cdot)$: Midpoint
- Vol$(\cdot)$: Box volume
- $p_T(\cdot)$: Polynomial part of interval Taylor model
- $[i]$: Remainder part of interval Taylor model
- $Rg(\cdot)$: Range of interval Taylor model
- $L[x]$: Left children of box
- $R[x]$: Right children of box
- $\Xi$: Over-approximation of set
- $\mathcal{X}$: Under-approximation of set
- $[x_0]$: Root of subpaving
- $test$: Test function
- $C_S$: Contractor
- $\mathcal{X}^c$: Complementary set
Overestimation

$\textit{oe}_{sp}$ \hspace{20pt} Overestimation caused by wrapping effect

$\textit{oe}_{\text{test}}$ \hspace{20pt} Overestimation caused by test function

$\textit{oe}_{\text{ct}}$ \hspace{20pt} Overestimation caused by contractor

$\mathcal{C}$ \hspace{20pt} Time independent capture basin

$T^-(\cdot)$ \hspace{20pt} Under-approximation of backward reachable set

$T^+(\cdot)$ \hspace{20pt} Over-approximation of backward reachable set

$\varphi_{\text{inv}}(\cdot;\cdot;\cdot)$ \hspace{20pt} Backward trajectory

$\mathfrak{A}$ \hspace{20pt} Set of samples of control input

$\textit{oe}_{\text{envelope}}$ \hspace{20pt} Overestimation caused by the guaranteed integration method

$\textit{oe}_{\text{assump}}$ \hspace{20pt} Overestimation caused by assumption

**Acronyms**

- **SFE**: Safe Flight Envelope
- **LOC**: Loss of Control
- **LOS**: Loss of Separation
- **ATM**: Air Traffic Management
- **RoA**: Region of Attraction
- **SOS**: Sum-of Squares
- **IVP**: Initial Value Problem
- **PDE**: Partial Differential Equation
- **ODE**: Ordinary Differential Equation
- **HJI**: Hamilton-Jacobi-Isaacs
- **ITM**: Interval Taylor Model
- **VSPODE**: Validated Solver for Parametric ODEs
- **WENO**: Weighted Essentially non-oscillatory
- **LF**: Lax-Friedrichs
- **TVD**: Total Variation Diminishing
- **RK**: Runge-Kutta
- **RDA**: Remainder Differential Algebra
- **SIVIA**: Set Inverter Via Interval Analysis
- **B&P**: Branch and Prune
- **B&C**: Branch and Contract
- **NB**: Number of Boxes
- **MBS**: Minimum Box Size
Chapter 1: Introduction

1.1 Background

Backward reachable sets are the initial set of states of a dynamical system from where the system can be controlled to reach a given target set within a given time. Forward reachable sets are the sets of states that can be reached with some control input from a given initial set within a given time. Determining an Inner bound of a set $S$ amounts to compute a set contained in the set $S$ and determining an outer bound is computing a set that contains the set $S$.

There are many possible applications in aviation for a method that computes guaranteed bounds for reachable sets. One possible application, which uses the guaranteed inner bound, is to determine the Safe Flight Envelope (SFE). By prohibiting an airplane to leave the SFE, Loss-of-Control (LOC) can be prevented. LOC occurs when the pilot or auto-pilot is not able to return to the normal flight envelope from its current flight condition. It is known that during the period between 1997 and 2006 LOC was the number two identified consequence on all fatal flight accidents (38.9%) preceded only by post-crash fire (42.4%). If we include non-fatal accidents, it was the number one cause (43%) of all accidents on the same period (Ranter, 2007). The SFE can be defined as the flight conditions from where the airplane can return to a trimmed condition. If there is a change in the dynamical behavior of the aircraft, the SFE will differ from the one obtained during flight testing. To take into account the variations of dynamical behavior, fast and reliable algorithms are needed to determine the SFE.

Another possible application that, in this case, uses outer bounds, is to ensure separation between aircrafts so that there is always a possible action to avoid Loss of Separation (LOS) for any type of actions of another aircraft. This problem is sometimes called the game of two identical vehicles. We can define LOS as when the protected zones of two aircrafts overlap. The protected zones can be defined as virtual cylinders around the aircraft with prescribed radius and height. Ensuring aircraft separation is a fundamental aspect when implementing a free flight ATM architecture, which allows aircrafts to take direct routes instead of fixed airways. This architecture has several advantages such as reducing congestion in the skies and providing more economical flights. If we develop better ways to ensure separation we can either improve safety or increase capacity in the skies. We can ensure separation by determining the positions and flight conditions from where it is impossible to avoid overlapping of the protected zones for some control of one of the aircrafts (Tomlin, Mitchell, & Ghosh, 2001). Those positions and flight conditions can be determined by computing the outer bound of the reachable set of the system composed by the two airplanes.

One method of estimating reachable sets is simulating a single trajectory of the system at a time. However, this is a computationally expensive method if we are dealing with systems with many different state values and input signals. Also, using only simulations, it is possible to miss important but isolated trajectories that lead to unsafe sets. To avoid those problems, there are methods of estimating reachable sets capturing the behavior of entire groups of trajectories at once. This concept of capturing the behavior of groups of trajectories is present in works about Region of Attraction (RoA)
analysis and reachability for systems with limited energy inputs, solving a Sum-of-Squares (SOS) relaxation (Packard, Topcu, Seiler, & Balas, 2009). Another work (Djeridane & Lygeros, 2006) discusses the use of neural networks to approximate the solution of the Hamilton-Jacobi-Isaacs (HJI) PDE to obtain the evolution of reachable sets. A MATLAB™ level set toolbox which approximates HJI PDEs through finite differences was developed by Mitchell (Mitchell I., 2002).

To obtain guaranteed bounds on reachable sets one solution is to compute reachable sets in terms of wrappers instead of points. Guaranteed integration using Interval Analysis is a widely developed field (Neher, Jackson, & Nedialkov, 2007) which gives guaranteed solutions of Initial Value Problems (IVP). Interval analysis was applied also on several other aerospace related topics such as finding trim-points for nonlinear aircraft models (van Kampen, Chu, Mulder, & van Emden, 2007), pilot model identification (van Kampen, Zaal, de Weerdt, Chu, & Mulder, 2010), integer ambiguity resolution for aircraft attitude determination (van Kampen, de Weerdt, Chu, & Mulder, 2009) (de Weerdt, van Kampen, Chu, & Mulder, 2008), spacecraft re-entry optimization (Filipe, de Weerdt, van Kampen, Chu, & Mulder, 2009) and fuel optimization for constrained spacecraft formation rotations (de Weerdt, Chu, & Mulder, Global Fuel Optimization for Constrained Spacecraft Formation Rotations, 2009).

Although there has been quite some work on the field of computing inclusions of the solution to initial value problems (Nedialkov N., 2008), to compute reachable sets of a dynamical system it is necessary to consider the control and disturbance inputs as interval parameters, and the existent solvers result on a large over-approximation when interval parameters are used. There is still a deficit of methods that give guaranteed tight bounds of reachable sets, and the existing methods are not efficient solving high dimension problems. With this notion we can say that there can be much improvement in this area. This thesis aims to give some contribution to solve this issue.

1.2 Problem Definition

The problem of determining reachable sets for an autonomous system is equivalent to the IVP solved by guaranteed integration using Interval Analysis (Moore, Kearfott, & Cloud, 2009). A generalization for reachable sets for systems with bounded inputs can be made by considering the input as an interval constant, on the flux expression. At the moment, the available guaranteed integration solvers are not adequate when the function to integrate has an interval parameter, and when the problem is multidimensional. This is due to the dependency problems and the wrapping effect.

The use of subpavings to compute inner and outer bounds of capture basins when the final time tends to infinity ($t \to \infty$) was already discussed by Jaulin and others (Lhommeau, Jaulin, & Hardouin, 2011). State bounding for discrete systems through subpavings was discussed in the book of Luc Jaulin (Jaulin, Kiefer, Didrit, & Walter, 2001). In this work those techniques developed by Jaulin (Jaulin, Kiefer, Didrit, & Walter, 2001) will be used to compute inner and outer bounds of reachable sets for a finite final time $t$. 
1.3 Thesis goal

The objective of this work is to apply the theory of Interval Analysis to obtain guaranteed inner and outer bounds of reachable sets of dynamical systems. There should be special interest in computing inner bounds since, as was seen on past experiences, conservative algorithms can lead to small inner bounds of little practical interest.

The first step was to devise a method that use interval analysis to compute guaranteed inner and outer bounds of reachable sets. The obtained method uses an Eulerian perspective, since the solution is computed for fixed boxes in space, while other known algorithms that yield guaranteed bounds use Lagrangean methods, because the result is a product of state propagation.

After this first step, with the devised method the objective was to reduce the underestimation or overestimation of the results. In this phase special concern was given to the inner bounds, since it was observed that it is a challenging problem because, if the method is too conservative, the inner bound of the reachable set can stop its expansion too soon and give poor results.

To improve the method one can take into account that it is not necessary to evaluate the flow in the initial set because, in the trajectory that goes into the border of the reachable set, the system is initially (in the case of forward reachable sets) or in the end (in the case of backward reachable sets) on the border of the initial or target set. Also we can take into account that the Hamilton-Jacobi-Isaacs (HJI) PDE is hyperbolic and only upwind information is needed.

One big issue in level set methods is the ‘curse of dimensionality’ because the number of nodes grows exponentially with dimension, making it almost impossible to use them in high dimensions. On this work the intention is that, if possible, the devised methods or improvements work efficiently in high dimensions.

Finally, the devised method was tested on the practical problem of the double integrator, given its similarity with many systems in the real world, to demonstrate what was achieved.

1.4 Thesis Outline

This thesis is structured as follows. In chapter 2 the theory behind reachable sets is presented. First, on section 2.1 reachable sets are defined. On sections 2.2 and 2.3, the different alternative mathematical methods of describing the evolution of reachable sets are described. Next, on section 2.4, an exposition of some already existent methods of estimating reachable sets is given. The first presented method is a level set method developed by Ian Mitchell that uses finite differences to solve the HJI PDE. A method which solves the HJI PDE using a neural network is also presented. Finally, a method of studying the reachability of a system using an SOS relaxation is presented. On section 2.5 the conclusions from this chapter are presented.
In chapter 3 an introduction to the field of Interval Analysis and its implementations to the problem of guaranteed integration are given. At first, on section 3.1, a brief introduction to Interval Analysis is given. On section 3.2 an exposition of different methods of obtaining inclusion functions is given. After that, on section 3.3, an introduction to the field of guaranteed integration is expounded. The next section, 3.4, exposes the application of Interval Taylor Models (ITM) to guaranteed integration. On section 3.5 a revision of the existent solvers using Interval Analysis to compute guaranteed solutions of IVPs is presented. Section 3.6 exposes the theory behind subpavings, which allows the enclosure of sets of varied shapes, with Interval Analysis. The last section, 3.7, holds the conclusions that can be derived from this chapter.

Chapter 4 contains the theory behind the application of the methods developed by Luc Jaulin of generating subpavings contained by or containing a determined set (Jaulin, Kiefer, Didrit, & Walter, 2001), to give guaranteed inner and outer bounds of reachable sets. The first section of this chapter, section 4.1, summarizes an existent method of computing inner and outer bounds of a time independent capture basin, which is the reachable set when the final time tends to infinity. Section 4.2 presents the formulation associated with the developed method. The last section, 4.3, presents the conclusions from this chapter.

On chapter 5 the computational application of the developed method is described. On the first section, 5.1, an algorithm of creating inner and outer bounds of sets (Branch and Prune) is presented. On section 5.2 a Branch and Prune algorithm, more efficient for the problem of reachable sets is described. The next section, 5.3, describes some improvements to solve computational issues found on the algorithms described previously. Next, on section 5.4, a similar algorithm that uses contractors is described. Section 5.5 holds a description of the application of the developed algorithms to compute outer bounds on reachable sets. After that follows section 5.6 on the application to inner bounds of reachable sets. The application of the Euler method of performing the guaranteed integration is described on section 5.7. The application of Interval Taylor Models, with VSPODE, to the program is contained on section 5.8. The conclusions from this chapter are contained on the last section, section 5.9.

Chapter 6 holds the validation experiment results of the devised algorithms to compute inner and outer bounds. The algorithm is tested with the double integrator example. First, on section 6.1, the analytical solution is computed. On section 6.2 the results obtained with the Branch and Prune algorithm are shown. The next section, 6.3, contains the results obtained using the Branch and Contract algorithm. Section 6.4 contains the results obtained when using VSPODE to perform the guaranteed integration on the Branch and Contract algorithm. The last section of this chapter, section 6.5, contains the conclusions that can be obtained from this chapter.

In chapter 7 the conclusions and recommendations that can be derived from the work in this thesis are presented.
Chapter 2: The Problem - Reachable Sets

This chapter aims to describe the problem of computing reachable sets. Some different equivalent formulations of that problem will be exposed, namely the Hamilton-Jacobi-Isaacs PDE, the capture basin formulation and a formulation based on the time to reach function which can be transformed into a static Hamilton-Jacobi PDE. Some existent methods to compute approximations of reachable sets will be briefly described.

This chapter is organized as follows. On section 2.1 the definitions of backwards and forwards reachable sets are given and the particular case of the initial value problem and some related theory is expounded. On the next section, 2.2, the formulation of the Hamilton-Jacobi-Isaacs PDE is given. On the next section, 2.3, some equivalent mathematical definitions of reachable sets are given. On section 2.4 a brief description of some methods to compute approximations of reachable sets or solve similar problems are given, namely level set methods (Mitchell I. , 2002), neural approximation of PDE (Djeridane & Lygeros, 2006) and Sum-of-Squares (SOS) feasibility (Packard, Topcu, Seiler, & Balas, 2009). Finally section 2.5 contains the conclusions from this chapter.

2.1 Definition

The interest of computing reachable sets is related to the need of capturing the behavior of several trajectories at once instead of relying on the simulation of a trajectory at a time. This theory consists of determining which states can be reached by a particular system with known dynamics from a set of initial conditions (forwards reachable set), or from which initial conditions is it possible to reach a specified target set of states (backwards reachable set) as illustrated in Figure 2-1.

In this work we will consider non-autonomous dynamical systems with a control and a disturbance input. Such systems can be described by the following formulation:

\[
\begin{align*}
\dot{x} &= f(x(t), u(t), d(t)) \\
x(0) &\in S_0 \text{ (or } x(t_f) \in T_0), t \in [0,t_f]
\end{align*}
\]

where \(0 \leq t_f < \infty\), \(x \in \mathbb{R}^n\) is the state, \(u \in \mathbb{R}^m\) is the control input, \(d \in \mathbb{R}^p\) is the disturbance input, \(S_0\) is an initial set of states, and \(T_0\) is a target set of states. The spaces of admissible control input and disturbance input trajectories are denoted as the spaces of piecewise continuous functions \(\mathcal{U} = \{u(.) \in PC^0|u(t) \in \mathbb{U}, 0 \leq t \leq t_f\}\), and \(\mathcal{D} = \{d(.) \in PC^0|d(t) \in \mathbb{D}, 0 \leq t \leq t_f\}\) respectively. \(\mathbb{U}\) and \(\mathbb{D}\) are the sets bounding the control input and the disturbance input respectively.

The information pattern is specified by a strategy for the control input, which is a map \(\gamma: \mathcal{D} \rightarrow \mathcal{U}\) specifying the input signal as a function of the disturbance signal. In this work we will consider non-anticipative strategies defined as follows.
Formally, we can define the forward and backwards reachable sets as follows.

**Definition I.1** (Forwards Reachable Set). The forward reachable set $S(\tau)$ at time $\tau$ ($0 \leq \tau \leq \tau_f$), of the system (2-1) from the initial set $S_0$, is the set of all states $x(\tau)$ such that there exists a strategy $u(t) = \gamma[d](t) \in \mathcal{U}(\tau \leq t \leq \tau_f)$, for which $x(\tau)$ is reachable from some $x(0) \in S_0$, for all disturbance inputs $d(t) \in \mathcal{D}(\tau \leq t \leq \tau_f)$, along a trajectory satisfying (2-1).

**Definition I.2** (Backwards Reachable Set). The backward reachable set $T(\tau)$ at time $\tau$ ($0 \leq \tau \leq \tau_f$), of the system (2-1) from the target set $T_0$, is the set of all states $x(\tau)$ such that there exists a strategy $u(t) = \gamma[d](t) \in \mathcal{U}(\tau \leq t \leq \tau_f)$, for which some $x(\tau_f) \in T_0$ is reachable from $x(\tau)$, for all disturbance inputs $d(t) \in \mathcal{D}(\tau \leq t \leq \tau_f)$, along a trajectory satisfying (2-1).

This definition gives some advantage to the control input $u(t)$ since it can be selected after knowing the disturbance input $d(t)$ at a given moment, although when computing reachable sets we are only interested on the most effective disturbance input for a given strategy. The restriction to non-anticipative strategies signifies that the strategy determines $u(t)$ based only on the current disturbance and state.

![Figure 2-1: Backward (left) and Forward (right) reachable sets](image)

The same problem for an autonomous system can be described as the initial value problem (IVP). The IVP is the problem of ordinary differential equations (ODEs) subject to initial conditions. This problem is formulated as follows:

$$
\dot{x} = f(t, x(t)),
$$

$$
x(t_0) = x_0
$$

One important theorem that guarantees existence and uniqueness for the initial value problem is the Picard-Lindelöf theorem (Braun, 1993).
Picard-Lindelöf Theorem

Consider the initial value problem.

\[ \dot{x}(t) = f(t, x(t)), \]
\[ x(t_0) = x_0, \quad t \in [t_0 - \varepsilon, t_0 + \varepsilon] \]

(2-4)

Suppose \( f \) is Lipchitz continuous in \( x \) and continuous in \( t \). Then, for some value \( \varepsilon > 0 \), there exists a unique solution \( x(t) \) to the initial value problem within the range \([t_0 - \varepsilon, t_0 + \varepsilon]\).

The proof of this theorem uses the Picard iterations which are built as follows:

\[ \theta_0(t) = x_0 \]
\[ \text{and} \]
\[ \theta_i(t) = x_0 + \int_{t_0}^{t} f(s, \theta_{i-1}(s)) ds \]

(2-5)

Using the Banach fixed point theorem, it can be shown that the sequence of "Picard iterates" \( \theta_i(t) \) is convergent and that the limit is a solution to the problem. This property of the Picard iterates can be useful to calculate an approximate solution to an IVP.

2.2 Hamilton-Jacobi-Isaacs (HJI) PDE

A possible method, with proven results, for computing the reachable sets is expressing them as the zero sublevel set of a viscosity solution of the Hamilton-Jacobi-Isaacs (HJI) PDE (Mitchell I., 2002).

For the forward reachable set we have:

\[ S(\tau) = \{x: \varphi_f(x, \tau) \leq 0\} \]

(2-6)

Where \( \varphi_f(x, \tau) \) is determined by the following equation:

\[ \frac{\partial \varphi_f(x, \tau)}{\partial \tau} + \max_{u \in \mathbb{U}} \min_{d \in \mathbb{D}} \left( \frac{\partial \varphi_f(x, \tau)}{\partial x} \right)^T f(x, u, d) = 0 \]

(2-7)

Similarly for the backward reachable set, considering \( 0 \leq \tau \leq t_b \), can be expressed as follows:

\[ T(\tau) = \{x: \varphi_b(x, t_b - \tau) \leq 0\} \]

(2-8)

where \( \varphi_b(x, t) \) is a viscosity solution of the following HJI PDE:

\[ \frac{\partial \varphi_b(x, t)}{\partial t} - \min_{u \in \mathbb{U}} \max_{d \in \mathbb{D}} \left( \frac{\partial \varphi_b(x, t)}{\partial x} \right)^T f(x, u, d) = 0 \]

(2-9)
Finally, with the above definitions we can define the Safe Flight Envelope of an aircraft as $O = S(t_f) \cap T(0) = \{ x : \text{max} (\varphi_f(x, t_f), \varphi_b(x, t_b)) \leq 0 \}$, as is shown in Figure 2-2, where the initial and target sets are the set of trim conditions.

![Figure 2-2: Illustration of the definition of Safe Flight Envelope (Safe Maneuver Set)](image)

### 2.3 Alternative formulations

Currently, the methods to compute reachable sets can be divided by the use of the time dependent Hamilton-Jacobi formulation (level set method), the static Hamilton-Jacobi formulation or the viability theory (capture basin methods).

The minimum time to reach function is defined by the following expression:

$$ T(x) = \inf_{u \in U} t(x, u(\cdot)) \quad (2.11) $$

In the previous expression, the time to reach function gives the minimum time to reach a state $x$ with the particular control input $u(\cdot)$, from an initial state on $T_0$. It is formally defined as follows.

$$ t(x, u(\cdot)) = \begin{cases} \min \{ t | \varphi(t, x, u(\cdot)) \in T_0 \}, & \text{if } \{ t | \varphi(t, x, u(\cdot)) \in T_0 \} \neq \emptyset \\ + \infty, & \text{otherwise} \end{cases} \quad (2.12) $$

The trajectory $\varphi(t, x_0, u(\cdot))$ is the solution $x(t)$ of the initial value problem $\dot{x}(t) = f(x, u(t))$ for the initial state $x_0$.

In short, the minimum time to reach function is the minimum possible time to reach a determined point using the control input. Practical implementations of this formulation work with the discounted minimum time to reach function $TD(x)$, which is the Kružkov transform of $T(x)$.

$$ TD(x) = 1 - e^{-T(x)} \quad (2.13) $$

$TD(x)$ is the solution to the following static Hamilton-Jacobi equation:

$$ TD(x) + H \left( x, \frac{\partial TD(x)}{\partial x} \right) + 1 = 0 \quad \text{for } x \in T_0^c \quad (2.14) $$

$$ TD(x) = 0 \quad \text{for } x \in \partial T_0 \quad (2.15) $$
The Hamiltonian $H \left( x, \frac{\partial \phi_\tau (x)}{\partial x} \right)$ is defined in the following way:

$$H(x,p) = \min_{u \in U} p^T f(x,u)$$  \hspace{1cm} (2-16)

Equations (2-13), (2-14), (2-15) and (2-16) showed how defining the backwards reachable set as $\mathcal{S}(\tau) = \{x \in \mathbb{R}^n | T(x) \leq \tau\}$ and applying the Kružkov transform (2-13), computing the forwards reachable set amounts to solve a static Hamilton-Jacobi PDE.

Finally, the capture basin $Capt_\tau(T_0, \tau)$ is defined as:

$$Capt_\tau(T_0, \tau) \triangleq \{x \in \mathbb{R}^n | \exists u(\cdot) \in U, \exists t \in [0, \tau], \phi(t, x, u(\cdot)) \in T_0\}$$  \hspace{1cm} (2-17)

In words, the capture basin is the set of states from which emanate at least one trajectory leading to $T_0$ in time $\tau$ or less.

The equivalence of the three formulations is stated in the following theorem, where for simplicity we consider systems with a control input and without a disturbance input. This theory can be read in more detail in (Mitchell I., 2002).

**Theorem I** (Equivalence of different formulations)
If the set of initial states contains only equilibrium points, the following sets are equivalent.

$$\mathcal{T}(0) = Capt_\tau(T_0, t_f) = \{x \in \mathbb{R}^n | T(x) \leq t_f\} = \{x \in \mathbb{R}^n | \phi_\tau(x, t_f) \leq 0\}$$  \hspace{1cm} (2-10)

On the last theorem $\mathcal{T}(0)$ is the backwards reachable set defined previously.

Similarly, the previous formulations could also be derived for forward reachable sets. For the sake of brevity those formulations will not be described here.

On the next section we will see some of the methods used currently to compute reachable sets and discuss their characteristics.

### 2.4 Available methods

**Level set methods**

The use of level set methods to compute reachable sets is thoroughly discussed in the PhD dissertation of Ian Michael Mitchell of the Stanford University that was submitted in 2002 (Mitchell I., 2002). This work explores the use of level set methods by discretizing the HJI PDE through finite differences. Specifically, it was experimented using a *weighted essentially non-oscillatory* (WENO) Lax-Friedrichs (LF) scheme fifth order accurate for the Hamiltonian $\max_{u \in U} \min_{d \in D} \left( \frac{\partial \phi_\tau (x,d)}{\partial x} \right)^T f(x,u,d)$ and *total variation diminishing* (TVD) Runge-Kutta (RK) schemes for the time derivative. Other discretizations are also possible.
The algorithm was tested with the pursuer-evader problem and yielded the results on Figure 2-3. The pursuer evader model is a model that describes a game of two vehicles with constant linear velocities ($v_p$ for the pursuer and $v_e$ for the evader), where the angular velocity of both vehicles is controlled. The angular velocity of the pursuer is considered the disturbance input and the evader angular velocity is the control input. The pursuer’s relative position and heading is described by the following system:

$$\dot{x} = \begin{bmatrix} \dot{x}_r \\ \dot{y}_r \\ \dot{\psi}_r \end{bmatrix} = \begin{bmatrix} -v_e + v_p \cos(\psi_r) + u y_r \\ v_e \sin(\psi_r) - u x_r \\ d - u \end{bmatrix} = f(x, u, d) \quad (2-18)$$

The results in Figure 2-3 are obtained with a target set defined by a circle around the evader defined by $\sqrt{x_r^2 + y_r^2} \leq 1$, the linear velocities are equal and set to 5 ($v_p = v_e = 5$) and the controls are limited by $U = D = [-1, 1]$. On Figure 2-3 the set is represented on the coordinates $(x_r, y_r, \psi_r)$ with the base representing $\psi_r = 0$ and the top representing $\psi_r = 2$. The images from left to right are taken, respectively, from the positive $x_r$ axis, the positive $y_r$ axis, and the negative $y_r$ axis.

![Figure 2-3: Results of the pursuer-evader system reachable set from different angles](image)

This method is considered to be accurate by the author of (Mitchell I., 2002) when comparing to other formulations of the problem because, in the pursuer-evader problem, the algorithm resolved the boundary of the reachable set to less than one tenth of a grid cell in most of the state space. The author of (Mitchell I., 2002) also considers that another advantage over other methods is that it provides a signed distance function that gives the distance and direction to the nearest point on the boundary of the reachable set.

On the other hand, since it is a grid based method and misses some possible trajectories, it cannot provide guaranteed bounds. Also, it is likely to be slower than algorithms based on other formulations and its computational cost rises exponentially with dimension.

The use of level set methods to generate over-approximations of reachable sets is discussed in the paper (Mitchell & Tomlin, 2003) approved in revised form in October 13, 2002. That paper was developed by Ian M. Mitchell and Claire J. Tomlin and supported by DARPA. The paper (Mitchell & Tomlin, 2003) claims that it is possible to generate over-approximations of reachable sets by solving the Hamilton-Jacobi-Isaacs PDE on a projection of the state space in a lower dimension. This technique was tested also on the pursuer evader system yielding the results on Figure 2-4.
This technique has the advantage of reducing the dimensionality of the problem thus reducing the computational load. The result is an over approximation, and the author claims that it is possible to generate also under approximations using the transformation $z_i = \frac{x_i}{|x_i|^2}$. The problem with the projection technique is that outer or inner bounds are not guaranteed if we use finite differences to compute an approximation of the HJI PDE, since this technique uses grids to discretize the equation.

**Neural approximation of PDE**

The application of neural networks to determine reachable sets is discussed in the work developed by Badis Djeridane and John Lygeros (Djeridane & Lygeros, 2006), which was presented at the 45th IEEE Conference on Decision & Control in December 2006. In this work the set function is approximated by the following form.

$$V_p(x, t) = l(x) + (T - t)F(x, t, p)$$

(2-19)

$F(x, t, p)$ is a single output neural network with adjustable weights $p$. With this approximation the algorithm developed in this work optimizes the weights $p$ such that $V_p(x, t)$ gives the minimum value of the following cost function on some training points.

$$G(x, t, p) = \frac{\partial V_p}{\partial t}(x, t) + \min \left\{ 0, \sup_{u \in U} \frac{\partial V_p}{\partial x}(x, t)f(x, u) \right\}$$

(2-20)

The training points are chosen to be close to the borders of the initial set, because that is where there is interest to observe the evolution of $V_p(x, t)$, so the set function must be as accurate as possible in that region. Figure 2-5 shows the distribution of training data for an initial set of states given by $[180, 240] \times [-0.4, 0.4]$. 

---

**Figure 2-4: Results of the pursuer-evader system reachable set from different angles**
Figure 2-5: Training points around the boundary of the initial set (rectangle)

This method was tested, using a smooth optimization algorithm, for a model representing the airspeed \( V \) and flight path angle \( \gamma \) of an aircraft. The dynamics of the model is given by the following differential equations.

\[
\begin{align*}
\dot{V} &= -\frac{\alpha_D V^2}{m} - g \sin(\gamma) + \frac{1}{m} u_1 \\
\dot{\gamma} &= \frac{\alpha_L (1-c')}{m} \frac{g \cos(\gamma)}{V} + \frac{\alpha_D c'}{m} u_2
\end{align*}
\]  

(2-21)  

(2-22)

The two control inputs are the thrust \( u_1 \in [T_{\text{min}}, T_{\text{max}}] \) and the aircraft pitch angle \( u_2 \in [\theta_{\text{min}}, \theta_{\text{max}}] \). The result displayed on Figure 2-6 is obtained using as parameters \( m = 85000kg \), \( c = 6 \), \( \alpha_L = 30 \), \( \alpha_D = 2 \), \( T_{\text{min}} = 40000N \), \( T_{\text{max}} = 80000N \) and \( \theta_{\text{min}} = -22.5^\circ \), \( \theta_{\text{max}} = 22.5^\circ \) and for a target set of \( (V, \gamma) \in [180m/s, 240m/s] \times [-0.4, 0.4] \).

Figure 2-6: Result from the neural approximation of the HJI PDE
The authors of this paper (Djeridane & Lygeros, 2006) believe that this method possibly will not require exponential increasing computational effort with dimension, as happens with other level set methods. If true, this method would help solve the ‘curse of dimensionality’ found on other methods to compute reachable sets. This method has some problems with convergence since it sometimes gets stuck in a local minimum or is slow in the final stages of convergence. Also it is computationally expensive due to matrix storage and updating requirements. This problem can probably be circumvented with more efficient algorithms. Also this method does not give guaranteed bounds since it only solves the HJI PDE in a finite set of points.

**SOS feasibility**

A reachability analysis is formulated as a sum-of squares feasibility problem on a document written before October 2009 by professors from the University Of California, California Institute of Technology and the University of Minnesota (Packard, Topcu, Seiler, & Balas, 2009). In that work is discussed mainly the computation of the region of attraction of a system. There, the reachable set is defined in a different way of the used on this work. In (Packard, Topcu, Seiler, & Balas, 2009) the $L_2$ norm of the input is limited. The $L_2$ norm is defined as follows.

\[
\|u\|_{2,T} = \sqrt{\int_0^T u(t)^T u(t) dt}
\]  

For simplicity we can denote $\|u\|_{2,T}$ as $\|u\|_2$. The reachable set is then defined as follows.

\[
\text{Reach}_R := \{x(t): x(0) = 0, t \geq 0, \|w\|_2 \leq R\}
\]  

Where $x(t)$ is described by dynamics of the following form:

\[
\dot{x} = f(x, w) \quad \text{with} \quad f(0,0) = 0
\]  

The approach used in (Packard, Topcu, Seiler, & Balas, 2009) is to find functions $V$ that satisfy the conditions of the following theorem.

**Theorem II**

If there exists a continuously differentiable function $V$ such that:

- $V(x) > 0$ for all $x \neq 0$ and $V(0) = 0$
- $\Omega_{V,R^2} = \{\xi : V(\xi) \leq R^2\}$ is bounded
- $\nabla f(x, w) \leq w^T w$ for all $x \in \Omega_{V,R^2}$ and for all $w \in \mathbb{R}^{nw}$

then $\text{Reach}_R \subseteq \Omega_{V,R^2}$.

If we define $\Omega_{p,\beta} = \{\xi : p(\xi) \leq \beta\}$ as a set that should be included in the reachable set, the following optimization problem is obtained.
Given $\beta$ solve, $\max_{V, R^2} R^2$
Such that $\Omega_{V, R^2} \subseteq \Omega_{p, \beta}$
$V$ satisfies the conditions of theorem II.

The same optimization problem can be relaxed by reformulating it as the following problem where the same conditions are met.

$$\max_{V, R^2, s_1, s_2} R^2$$
subject to:

$$-[(\beta - p) + (V - R^2)s_2] \text{ is } \text{SOS}[x].$$
$$-[(R^2 - V)s_2 + \nabla V f(x, w) + w^T w] \text{ is } \text{SOS}[x, w].$$
$$V - \epsilon x^T x \text{ is } \text{SOS}[x], V(0) = 0, \text{ and }$$
$s_1, s_2, s_3$ are $\text{SOS}$

$\text{SOS}$ (sum of squares) functions are functions that can be expressed as sums of squares of polynomials. The advantage of using $\text{SOS}$ constraints is that we can transform them into $\text{LMI}$ (linear matrix inequality) constraints, simplifying the optimization. With the given formulation we can have guaranteed bounds on the reachable sets.

On the other hand the comparison with the other methods is difficult because other definitions of reachable sets do not count with the restriction $||w||_2 \leq R$.

2.5 Conclusions

In this chapter we have seen that there are at least four different formulations for reachable sets, which are the definitions I.1 or I.2 and the formulations based respectively on the HJI PDE, on the capture basin and on the time to reach function. Theorem I states the equivalence of those four formulations.

We have also seen in section 2.4 some published methods of computing reachable sets. The first two methods yield approximated results and not guaranteed results. The third method computes guaranteed results of reachable sets when the control has a limited $L_2$ norm, and on this work we are considering bounded controls but we do not impose constraints on the $L_2$ norm of the controls.

On the next chapter a short introduction to the field of interval analysis is given.

Chapter 3: The Approach - Interval Analysis

Interval analysis allows computations without an associated error. It represents the set of possible results as intervals, or in the n-dimensional space as the tensor product of $n$ intervals, here referred as a box. Interval analysis is also useful to perform global optimizations, since it is possible to evaluate functions in entire sets, instead of a collection points, and the result is guaranteed to be a global
optimum. In this thesis interval analysis will be used to perform tests on boxes of states, determining if a box is included or includes some part of a reachable set, thus capturing the behavior of all the trajectories beginning (or ending) in that box.

This chapter is organized as follows. In section 3.1 a brief introduction to the field of interval analysis is given. In the next section, 3.2, different techniques of evaluating functions on intervals will be explained. Section 3.3 describes a generalization of the Picard iterates to interval analysis to give guaranteed end results of IVPs. The next section, 3.4, describes the use of Interval Taylor Models which are useful to reduce the wrapping effect. Section 3.5 describes the theory of using unions of boxes to over- or under-approximate a set. Section 3.6 gives a summary of the developed programs until now. The last section, 3.7, contains the conclusions from this chapter.

3.1 Introduction to Interval Analysis

Grid based methods for computing reachable sets are not able to give guaranteed bounds, since it is possible to miss some important trajectory. One possible way to give guaranteed bounds is using interval analysis, which uses interval numbers instead of real numbers and extends some of the usual operations of real analysis into operations between intervals (Moore, Kearfott, & Cloud, 2009).

In this thesis an interval will be represented by $[x, \bar{x}]$, where $x$ is the lower bound and $\bar{x}$ the upper bound. An interval variable will be denoted by a lower case letter enclosed by brackets, $[x]$, where $x$ is usually a real variable enclosed by $[x]$. There will also be used interval vectors, which are vectors where each component is an interval, and will also be referred to as boxes. They will be denoted as $([x_1, \bar{x}_1], ..., [x_n, \bar{x}_n])$, where $[x_i, \bar{x}_i]$ are its components. Interval vector variables will also be represented by lower case letters enclosed by brackets $[x]$ and its $i^{th}$ component as $[x_i]$. On this thesis we will also deal with interval matrices, usually represented by capital letters, for instance $A$, and the component from its $j^{th}$ row and $i^{th}$ column is then denoted as $[a_{ij}]$. The set of real intervals is referred to by the symbol $\mathbb{I}\mathbb{R}$. Accordingly, the set of interval vectors of dimension $n$ is referred to as $\mathbb{I}\mathbb{R}^n$, and the set of all interval matrices with $n$ rows and $m$ columns is denoted as $\mathbb{I}\mathbb{R}^{n \times m}$. For a given set $S \subset \mathbb{R}^n$ the notation $[S]$ will be used to denote the interval vector with smaller volume such that $[S] \supseteq S$.

Two important characteristics of an interval $[x]$ are its width $w([x])$ and midpoint $m([x])$ defined by equations (3-1) and (3-2) respectively.

$$w([x]) = \bar{x} - x$$  \hspace{1cm} (3-1)

$$m([x]) = \frac{\bar{x} + x}{2}$$  \hspace{1cm} (3-2)

For interval vectors the width, midpoint and volume are defined as follows:

$$w([x]) = \max_i w([x_i])$$  \hspace{1cm} (3-3)
Using interval analysis we can search for inclusion functions \([f]([x])\), from intervals to intervals, such that if \(x\) is contained in \([x]\), \(f(x)\) is contained in \([f]([x])\). One property of an inclusion function is that it can be inclusion isotonic (if \([x] \subseteq [y]\) then \([f]([x]) \subseteq [f]([y])\)), as is expressed on theorem III. First we need to define rational interval functions.

The interval arithmetic operations are an extension to intervals of the arithmetic operations of real numbers, namely the sum, subtraction, multiplication and division, and are computed as expressed in equations (3-6) to (3-9).

\[
(a, b) + [c, d] = [a + c, b + d]
\]

\[
(a, b) - [c, d] = [a - c, b - d]
\]

\[
(a, b) \times [c, d] = [\min(ab, ac, ad, bd), \max(ab, ac, ad, bd)]
\]

\[
\frac{[a,b]}{[c,d]} = [a,b] \times [1/d, 1/c], \text{ if } 0 \notin [c,d]
\]

The rational interval functions are then a subset of the interval valued functions defined according to definition II.

**Definition II.** A rational interval function is an interval-valued function whose values are defined by a specific sequence of interval arithmetic operations.

The next theorem holds for rational inclusion functions but it could be extended for various other functions well defined in interval analysis such as \(\sin([x])\) or \(\ln([x])\).

**Theorem III**

If \([f]([x_1],[x_2],\ldots,[x_n])\) is a rational expression in the interval variables \([x_1],[x_2],\ldots,[x_n]\) and a finite set of constant intervals with interval arithmetic operations, then

\([x_1] \subseteq [x_2] \subseteq \ldots \subseteq [x_n]\)

implies

\([f]([x_1],[x_2],\ldots,[x_n]) \subseteq [f]([x_1],[x_2],\ldots,[x_n])\)

for every set of interval numbers \([x_1],[x_2],\ldots,[x_n]\) for which the interval arithmetic operations in \([f]\) are defined.

From this theorem we can conclude that for smaller intervals where the inclusion function is evaluated \([x]\), the estimation on \(f(x)\) provided by the inclusion function \([f]([x])\) is tighter.
If we consider the example of Figure 3-1 where \( f(x) = x^2 \) and we take the inclusion function 
\[ f([x]) = [x] \cdot [x] \] and the interval \([-1,1]\) we can see that 
\[ f([-1,1]) = [-1,1] \] and the image set is 
\[ f([-1,1]) = [0,1] \] 
\( f([x]) = (y = f(x)|x \in [x]) \). In this case we have a dependency issue because by 
considering the inclusion function \([x] \cdot [x]\) every possible product on the interval \([-1,1]\) is considered 
and not just the products of one number by itself. This dependency issue disappears if \( f(x) \) is a real 
rational expression where each variable \( x_i \) appears only once.

![Figure 3-1: Example of the dependency problem](image1)

Another source of overestimation occurs if the result is not representable by an interval. In this case 
we have the wrapping effect. One example where this effect is evident (Figure 3-2) occurs if we 
consider the function \( f(x,y) = (x + y, y) \) and its natural extension with the same expression, 
\[ f([x],[y]) = ([x] + [y], [y]) \] evaluated over \([x] = [-1,1]\) and \([y] = [-1,1]\). The image set \( f([x],[y]) \) is 
a parallelogram with corners at \((-2, -1), (0, -1), (2, 1)\) and \((0, 1)\) whereas the inclusion function gives 
as result the box \( f([x],[y]) = ([2,2],[-1,1]) \).

![Figure 3-2: Example of the wrapping effect](image2)

For practical implementations of interval analysis there are several libraries, in many different 
programming languages, that create a new interval variable type and define the basic functions on 
interval variables. Two of the most popular libraries that implement interval analysis are PROFIL/BIAS 
(Knüppel, 1999) for C++ and INTLAB for MATLAB™ (Rump, 1999).

Finally, when dealing with finite precision, another problem arises. There is a basic requirement that 
the result of a computation with finite precision encloses the real result. To satisfy this requirement, a
library must make use of outward rounding, which consists of performing rounding towards $-\infty$ for the lower bound of an interval and rounding towards $+\infty$ for the upper bound.

### 3.2 Inclusion functions

Inclusion functions can have many forms such as the interval extension, the centered form, the mean value form, the slope form and the Taylor expansion (Moore, Kearfott, & Cloud, 2009). The simplest way to have an inclusion function is to substitute the operations in the expression of $f(x)$ with their equivalent on interval arithmetic. In this case we have an *interval extension*. In the *centered form* we write $f(x)$ in the following form.

$$f(x_1, \ldots, x_n) = f(c_1, \ldots, c_n) + g(y_1, \ldots, y_n) \quad (3-10)$$

with $y_i = x_i - c_i$. If $f(x)$ is rational we can write $g(y_1, \ldots, y_n)$ as follows.

$$g(y_1, \ldots, y_n) = y_1 h_1(y_1, \ldots, y_n) + \cdots + y_n h_n(y_1, \ldots, y_n) \quad (3-11)$$

where $h_i$ is rational and $h_i(0, \ldots, 0)$ is defined if $f(c_1, \ldots, c_n)$ is also defined. Then the inclusion function $[f_i]$ is defined as follows.

$$[f_i](y_1, \ldots, y_n) = f(c_1, \ldots, c_n) + \sum_{i=1}^n [y_i] \cdot [h_i]([y_1], \ldots, [y_n]) \quad (3-12)$$

where $c_i = m([x_i])$, $[y_i] = [x_i] - c_i$ and $[h_i]$ is the natural interval extension of $h_i$. In the mean value form we define $m$ as the mean value of the interval vector $[x]$ and $[d_i f]/[x]$ as the interval extension of $\partial f(x)/\partial x_i$. Then we define the *mean value extension* of $f$ on $[x]$ as follows.

$$[f_{me}](x) = f(m) + \sum_{i=1}^n [d_i f]/[x]([x] - m_i) \quad (3-13)$$

There is a more efficient form of the mean value extension described by Jaulin (Jaulin, Kiefer, Didrit, & Walter, 2001), the *mixed centered inclusion function*. This form reduces overestimation by reducing the area where the gradient is evaluated. In this case the interval extension is computed as follows.

$$[f_{mc}](x) = f(m) + \sum_{i=1}^n [d_i f]([x], [x_i], \ldots, [x_i], m_{i+1}, \ldots, m_n)([x] - m_i) \quad (3-14)$$

For the *slope form* we first have to define the slope function as follows.

$$f(y) - f(x) = s(f, x, y)(y - x) \quad (3-15)$$
For multivariate functions, the function \( s_i(f, x_1, \ldots, x_n, y) \) is the slope in the direction of \( x_i \). If we define \([s_i](f, [x], y)\) as an inclusion function of \( s_i(f, x_1, \ldots, x_n, y) \) we obtain the following inclusion function of \( f(x) \).

\[
[f_s](x, y) = f(y) + \sum_{i=1}^n [x_i] \cdot [s_i](f, [x], y)
\]  

(3-16)

Usually the slope form gives tighter bounds than the mean value form. There is also the monotonicity test form which simplifies the inclusion function for monotonic functions, and the Skelboe-Moore algorithm which computes bounds with prescribed accuracy performing a global optimization. Those methods are described in detail in (Moore, Kearfott, & Cloud, 2009).

To better illustrate the differences between the slope form and the mean value form we can look at the example of the function \( f(x) = x^2 \) with \([x] = [-1, 1]\), where we observe from Figure 3-3 and Figure 3-4 that \([f_{mv}](x) = [-2, 2]\) and \([f_s](x, 0) = [-1, 1]\).

![Figure 3-3: Example of mean value form](image1)

![Figure 3-4: Example of slope form](image2)

The Taylor expansion works for functions with one variable (here considered as \( t \)). We define the \( k^{th} \) Taylor coefficient \((x)_k\) as follows.
\( (x)_0 = x(t_0) \) \hfill (3-17)

\( (x)_k = \frac{1}{k!} \frac{d^k x}{dt^k} (t_0) \) \quad (k = 1, 2, \ldots) \hfill (3-18)

If \( R_N \) is an interval enclosure of \((x)_N(t)\) and, if it is in the radius of convergence of the Taylor expansion of \(x(t)\), we have the following enclosure of \(x(t)\):

\[
x(t) \in \sum_{k=0}^{N} \frac{1}{k!} (x)_k (t - t_0)^k + R_N([t_0, t])(t - t_0)^N
\hfill (3-19)
\]

The Taylor expansion can also give an enclosure of the solution of an IVP. For \( k \geq 1 \) the Taylor coefficients of \( y(t) \) are given by the following expression.

\[
(y)_k = \frac{1}{k} (f)_{k-1} (y)
\hfill (3-20)
\]

where the derivatives are evaluated at \(y(t_0)\) and \(R_N\) is evaluated at an interval that contains \(y(s)\) for \(s \in [t_0, t]\). If \(f\) is a combination of certain functions and arithmetic operations, the task of differentiating \(f(y)\) with respect to time is greatly simplified by using recursion relations such as:

\[
(u^a)_k = \left( \frac{1}{a} \right) \sum_{j=0}^{k-1} \left( a - \frac{j(a+1)}{k} \right) (u)_k - (u^a)_j,
\hfill (3-21)
\]

\[
(e^a)_k = \sum_{j=0}^{k-1} \left( 1 - \frac{j}{k} \right) (e^a)_j (u)_k - (e^a)_j,
\hfill (3-22)
\]

\[
(ln u)_k = \left( \frac{1}{u} \right) \left( (u)_k - \sum_{j=0}^{k-1} \left( 1 - \frac{j}{k} \right) (u)(ln u)_j \right),
\hfill (3-23)
\]

\[
(sin u)_k = \left( \frac{1}{u} \right) \sum_{j=0}^{k-1} (j + 1) (cos u)_{k-1-j} (u)_{j+1},
\hfill (3-24)
\]

\[
(cos u)_k = -\left( \frac{1}{u} \right) \sum_{j=0}^{k-1} (j + 1) (sin u)_{k-1-j} (u)_{j+1}.
\hfill (3-25)
\]

This principle is used in a programming technique called automatic differentiation, which consists of differentiating programs, creating new programs. The derived program gives as output the value of the derivative of the function defined by the first program.

### 3.3 Guaranteed Integration

It is also possible to integrate interval functions in the same way we integrate real valued functions (Moore, Kearfott, & Cloud, 2009). For an interval valued function \([f]([x])\) and \([x_i]\) a uniform subdivision of \([x] = [a, b]\) we can define finite sum \(S_N\).

\[
S_N ([f]; [a, b]) = \sum_{i=1}^{N} [f]([x_i])(b - a)/N
\hfill (3-26)
\]

Then we can define the integral of \([f]([x])\) as follows.
In practice, one possible way to compute the integral of an interval function is to use the above definition stopping at a finite \( N \).

Using the above method to perform integrations, it is possible to get an inclusion of the solution of an initial value problem if we perform Picard iterations. The homogeneous initial value problem, \( \dot{y}(t) = f(y) \) and \( y(0) = y_0 \), is formulated as the following integral equation.

\[
y(t) = p(y)(t)
\]  

where \( p(y)(t) \) is defined as follows.

\[
p(y)(t) = y_0 + \int_{[0, t]} f(y(s)) ds
\]  

In (3-28) it is assumed that \( y \) belongs to some class of real valued functions denoted as \( M_r \). In the rest of this section \( [x] \) and \( [y] \) will be considered to belong to a class \( M \) of interval enclosures of elements of \( M_r \). To compare members of the class \( M \) we have \( [x] \subseteq [y] \) if \( [x](t) \subseteq [y](t) \) for all \( t \) (in the common domain).

\([p]\) is an interval majorant of the functional \( p \) if \( p(y) \in [p]([y]) \) for \( y \in [y] \). The interval operator \([p]\) is inclusion isotonic if \( [x] \subseteq [y] \) implies \( [p]([x]) \subseteq [p]([y]) \). If \([f]\) is an inclusion isotonic interval enclosure of \( f \), the following interval operator is inclusion isotonic.

\[
[p]([y])(t) = y_0 + \int_{[0, t]} [f]([y]) ds
\]  

We can therefore use the following theorem to compute inclusion functions of \( y(t) \).

**Theorem IV**

*If \([p]\) is an inclusion isotonic interval majorant of \( p \), and if \([p]([y]^{(0)}) \subseteq [y]^{(0)} \), then the sequence defined by

\[
[y]^{(k+1)} = [p]([y]^{(k)}) \text{ for } (k = 0,1,2,\ldots)
\]  

has the following properties:

- \([y]^{(k+1)} \subseteq [y]^{(k)} \), \( k = 0,1,2,\ldots \)

- For every \( a \leq t \leq b \), the limit

\[
[y](t) = \bigcap_{k=0}^{\infty} [y]^{(k)}(t)
\]  

exists as an interval \([y](t) \subseteq [y]^{(k)}(t), k = 0,1,2,\ldots\)
• Any solution of (3-28) which is in \([y]^{(0)}\) is also in \([y]^{(k)}\) for all \(k\) and in \([y]\) as well. That is, if 
  \(y(t) \in [y]^{(0)}(t)\) for all \(a \leq t \leq b\), then \(y(t) \in [y]^{(k)}(t)\) for all \(k\) and all \(a \leq t \leq b\);

• If there is a real number \(c\) such that \(0 \leq c < 1\), for which \([x] \subseteq [y]^{(0)}\) implies
  \[ \sup_t w([p]([x])(t)) < c \sup_t w([x](t)), \quad a \leq t \leq b \]
  for every \([x] \in M\), then (3-28) has the unique solution \([y](t)\) in \([y]^{(0)}\) given by (3-32).

With this method it is possible to get an inclusion function of the initial value problem solution, assuming an uncertainty or perturbation in the initial value. Due to the dependency issue already discussed, the width of \([y](t)\) is always growing even if the solutions of the IVP are contracting. A solution of this problem will be discussed in section 3.5.

3.4 Interval Taylor Models

Interval Taylor Models (ITMs) are a more accurate way of enclose a set than boxes which combine interval arithmetic with symbolic computations. For some existing solvers of IVPs, namely COSY Infinity and VSPODE (Neher M., 2005), the computations are performed on ITMs instead of boxes. ITMs can also be used as interval extensions, as was seen in section 3.2.

ITMs are composed by a polynomial part, where the symbolic computations are performed, and a remainder part represented by an interval. Denoting the ITM as \(\mathcal{U}\), the polynomial part, a multivariate polynomial of order \(n\) in \(m\) variables, as \(p_n(x)\), the interval part as \([i]\), and a box \([x] \in \mathbb{R}^m\), the ITM is defined as follows:

\[ \mathcal{U} := p_n(x) + [i], \quad x \in [x] \quad (3-33) \]

For practical implementations the coefficients of the polynomial part \(p_n(x)\) must be representable as a floating point number, and if during the computations a coefficient must be rounded, the set of all possible roundoff errors must be added to the remainder part \([i]\).

In some cases it is convenient to translate intervals to ITMs. It is possible to do so using a vector \(c \in \mathbb{R}^m\), a positive diagonal matrix \(C \in \mathbb{R}^{m \times m}\) and a box \([x] \in \mathbb{R}^m\).

\[ \mathcal{U} := c + Cx, \quad x \in [x] \quad (3-34) \]

For an interval \([z]\) a convenient choice for \(c, C\) and \([x]\) is \(c = m([z]), C = \text{diag} \left( \frac{1}{2} w([z]) \right) \) and \([x] = ([−1,1], ..., [−1,1])\).

It was implied earlier that an ITM has a dual nature. An ITM can be interpreted as an inclusion function where (3-35) holds for \(x \in [x]\).

\[ f(x) \in p_n(x) + [i] \quad (3-35) \]
Also, an ITM can be interpreted as a form of representing the set defined by the range of \( U, Rg(U) \), defined by (3-36).

\[
Rg(U) = \{ z = p_n(x) + i | x \in [x], i \in [i] \} \tag{3-36}
\]

One of the practical interests of using ITMs is that they have an associated remainder differential algebra (RDA) that defines operations on ITMs. The RDA has been implemented computationally on COSY Infinity and VSPODE (Lin & Stadtherr, 2006). Considering two Taylor models of the same dimension \( p \) and order \( n \):

\[
U_1 := p_n^1(x) + [i]_1, \ x \in [x] \tag{3-37}
\]
\[
U_2 := p_n^2(x) + [i]_2, \ x \in [x] \tag{3-38}
\]

The RDA is defined in such a way that if \( f(x) \in U_1(x) \), and \( g(x) \in U_2(x) \) for \( x \in [x] \), then, if \( \ast \) is an operation defined by the RDA then \( f(x) \ast g(x) \in U_1 \ast U_2 \) for \( x \in [x] \). For example the sum and subtraction are defined as follows:

\[
U_3 = U_1 \pm U_2 = p_n^3(x) + [x]_3, \ x \in [x] \tag{3-39}
\]
\[
p_n^3(x) = p_n^1(x) \pm p_n^2(x) \tag{3-40}
\]
\[
[i]_3 = [i]_1 \pm [i]_2 \tag{3-41}
\]

If we consider the terms of order smaller or equal to \( n \) of \( p_n^1(x) \times p_n^2(x) \) as \( n(x) \) and the terms greater than \( n \) as \( p_g(x) \), the multiplication is defined by the RDA as follows:

\[
U_3 = U_1 \times U_2 = p_n^3(x) + [i]_3, \ x \in [x] \tag{3-42}
\]
\[
p_n^3(x) = n(x) \tag{3-43}
\]
\[
[i]_3 = p_g([x]) + p_n^1([x]) \times [i]_2 + p_n^2([x]) \times [i]_1 + [i]_1 \times [i]_2 \tag{3-44}
\]

Also composition of functions is defined if the dimension of the ITM \( U_2 \) is the same as the dimension of the variable \( x \), that is, \( p = m \), and \( Rg(U_2) \subset [x] \). On that case if we define \( p_s(x) \) as the terms of order smaller or equal than \( n \) of \( p_n^1(p_n^2(x)) \), and \( p_g(x) \) as its terms order greater than \( n \), the composition is defined as follows.

\[
U_3 = U_1 \circ U_2 = p_n^3(x) + [i]_3, \ x \in [x] \tag{3-45}
\]
\[
p_n^3(x) = p_s(x) \tag{3-46}
\]
\[
[i]_3 = p_g([x]) + p_n^1([i]_2) + [i]_1 \tag{3-47}
\]

With this definition, if \( f(x) \in U_1(x) \), and \( g(x) \in U_2(x) \) for \( x \in [x] \), then \( f(g(x)) \in U_1 \circ U_2 \) for \( x \in [x] \). Defining the composite function is very useful given the dual nature of ITMs. The Taylor model
represented by $\mathcal{U}_1 \subseteq \mathcal{U}_2$ can be interpreted as the evaluation of the interval extension of a function represented by $\mathcal{U}_1$ over the set enclosed by $Rg(\mathcal{U}_2)$.

To illustrate the advantages of using interval Taylor models we present here in Figure 3-5 and Figure 3-6 an example from (Berz & Hoffstätter, 1998) with the bounding of a function of two variables where it was observed that “the algorithm describing the function had to be evaluated at only one Taylor model to get the tightest bound shown whereas 6400 evaluations were needed to achieve the tightest bound computed by conventional interval arithmetic”.

![Figure 3-5: Interval bounding of a function by a Taylor model: orders 7 (left) and 10 (right)](image1)

![Figure 3-6: Conventional interval bounding of a function: from top left to bottom right the number of interval evaluations is 100, 400, 1600 and 6400](image2)

### 3.5 Available Solvers

A summary of the available methods that give inclusions of the solution to the IVP problem, and a description the latest version of the guaranteed integration program VNODE, VNODE-LP were on the slides of Nedialkov (Nedialkov N., 2008), from the Department of Computing and Software of the McMaster University, in 2008. The first known method (from 1988) using interval analysis for computing solutions of the IVP problem was AWA from R. Lohner. It uses a simple Taylor expansion for integration in time of first order, $k = 1$. The other programs VNODE, COSY and VSPODE use Taylor expansions for the time integration that help reduce the dependency problem present in the traditional methods.

The mentioned solvers use another method of guaranteed integration than the one described in section 3.3, the *mean-value evaluation*, which can follow contractions. It is described in the slides of Nedialkov (Nedialkov N., 2008) in Appendix A.
For VSPODE the mean value evaluation is extended to interval Taylor models (Lin & Stadtherr, 2006). A description of this method is as follows. The ODE to solve is of the type $y(t) = f(y(t))$ subject to $y(0) \in [y^{(0)}]$. The time step is denoted as $h_j = t_{j+1} - t_j$. A solution at time $t_j$ is guaranteed to belong to the set defined by the ITM $U_j := p_u, f(x) + [i]_j, x \in [x]$, that is, $y(t_j) \in Rg(U_j)$ for all $y(0) \in [y^{(0)}]$. The solution at time $t_{j+1}$, $U_{j+1}$, is computed given the ITM at time $t_j$, $U_j$. The integration in time is computed using the method of equations (3-19) and (3-20).

The first step is to enclose the solution of $y(t)$ for all possible trajectories between $t_j$ and $t_{j+1}$. That enclosure will be denoted as $[\hat{y}^{(0)}]$ and is computed as is expressed in equation (3-48).

$$[\hat{y}^{(0)}] = \sum_{l=0}^{k-1} [0, h_j]^i * [(f)_{i}([Rg(U_j)])] + [0, h_j]^k * [(f)_{k}([y^{(0)}])]} \quad (3-48)$$

In (3-48) $(f)_{i}(y)$ is the $i^{th}$ coefficient of the Taylor series expansion of $f(y)$ from (3-17) and (3-18), and can be computed as follows:

$$(f)_{0}(y) = y \quad (3-49)$$

$$(f)_{i}(y) = \frac{1}{i!} \left( \frac{\partial^{i} f}{\partial y^{i}} \right)(y), \text{ for } i \geq 1 \quad (3-50)$$

The integer $k$ represents the order of the Taylor expansion. The interval vector $[\hat{y}^{(0)}]$ is an initial overestimation of $[y^{(0)}]$, and therefore $[\hat{y}^{(0)}]$ is considered valid only if $[\hat{y}^{(0)}] \subseteq [y^{(0)}]$.

For the second step we compute $U_{j+1}$ from equations (3-52) and (3-53) using as inputs $U_j, f(y), [\hat{y}^{(0)}], A_j$ and $[r_i]$. Equation (3-53) shows how $[r_{j+1}]$ is computed. The ITM $U_{(f)i}$ represents a Taylor expansion of $(f)_{i}(y)$.

$$S_j = \left\{ \sum_{l=0}^{k-1} h_j^l \ast \left[ \frac{\partial(f)_i}{\partial y} \right] ([Rg(U_j)]) \right\} \quad (3-51)$$

$$U_{j+1} = p_u, f(x) + \sum_{l=0}^{k-1} h_j^l \ast U_{(f)i} \ast U_j + h_j^k \ast [(f)_{k}([y^{(0)}])] + (S_j A_j)[r_i], x \in [x] \quad (3-52)$$

$$[r_{j+1}] = \left\{ A_{j+1}^{-1}(S_j A_j))\right\}[r_i] \quad (3-53)$$

In (3-52) and (3-53), $[r_0] = [0,0]$ and $A_0 = I$ ($I$ is the identity matrix). $A_{j+1}$ can be selected as $A_{j+1} = m(S_j A_j)$, in this case we have the parallelepiped method, or we can choose $A_{j+1} = Q_{j+1}$ from the QR factorization $Q_{j+1}R_{j+1} = m(S_j A_j)$, this is the case of the Lohner’s QR method. Usually, Lohner’s QR method is preferred because in the parallelepiped method $A_j$ often becomes ill conditioned.

It is important to note that VSPODE and the other solvers admit interval parameters, and thus if we consider the control inputs as the interval parameters we can compute reachable sets.
3.6 Subpavings

A subpaving of a box $[x] \subset \mathbb{R}^n$ is a union of non-overlapping subboxes of $[x]$ with non-zero width. Non-overlapping means that the intersection of two such subboxes must be empty or have zero volume, that is, they may only intersect on their boundaries. Subpavings are useful to provide over- ($\overline{X}$) and under-approximations ($\underline{X}$) of a set $\underline{X} \subseteq [x]$ such that $\underline{X} \subseteq \overline{X} \subseteq [x]$. The theory associated with subpavings is thoroughly discussed in the book of Luc Jaulin (Jaulin, Kiefer, Didrit, & Walter, 2001). In this thesis only a brief description will be given.

A subpaving of $[x]$ can be obtained through successive bisections and eliminations, in that case it is called a regular subpaving. A bisection is a process of dividing a box into two smaller boxes of equal width. If we bisect a box $[x]$ it is called the parent, and the two smaller boxes are its left child $L[x]$ and its right child $R[x]$. The boxes $L[x]$ and $R[x]$ are called siblings. $L[x]$ and $R[x]$ are obtained as follows. Let's consider $j$ the index of the first component of $[x]$ of maximum width, that is, $j$ is defined from (3-54).

$$j = \min\{i | w([x_i]) = w([x])\}$$ (3-54)

The boxes $L[x]$ and $R[x]$ are then defined as follows:

$$L[x] \triangleq ([x_1, \ldots, x_j, m([x_j]), \ldots, [x_n])$$ (3-55)

$$R[x] \triangleq ([x_1, \ldots, [m([x_j)], \overline{x_j}], \ldots, [x_n])$$ (3-56)

From the above definitions we can also consider $L$ and $R$ as operations from $\mathbb{R}^n$ to $\mathbb{R}^n$.

A subpaving may be represented as a binary tree. A binary tree consists of a set of nodes. This set can be empty, have one node, on that case it is the root of the tree, or have two binary trees with an empty intersection, the left and right subtrees. A node other than the root of the tree may have a left child and a right child, and must have a parent of which it is children of. If a node has no children it is a degenerate node or a leaf. If two nodes have the same parent they are called siblings. A regular subpaving of a box $[x_0]$ is represented by a binary tree if we consider a node to represent a box $[x]$ and, if they exist on the binary tree, its left and right children represent respectively $L[x]$ and $R[x]$ as defined above, the root of the tree represents the initial box $[x_0]$. Figure 3-7 shows a representation of a binary tree and Figure 3-8 its corresponding subpaving on two dimensions.

![Figure 3-7: Binary tree](image-url)
A subpaving on a n-dimensional space may be viewed as a subset of \( \mathbb{R}^n \) or it may be viewed as a list of boxes, that is, a set of elements of \( \mathbb{R}^n \). To differentiate when referring to the subset of \( \mathbb{R}^n \) defined by the subpaving the notation \( \mathcal{Q} \) will be used, when referring to the list of boxes we will use calligraphic letters, in this case \( \mathcal{X} \).

To under- and over-approximate a set \( \mathcal{X} \) by two subpavings \( \overline{\mathcal{X}} \) and \( \bar{\mathcal{X}} \), \( \overline{\mathcal{X}} \subseteq \mathcal{X} \subseteq \bar{\mathcal{X}} \), an algorithm was developed by Jaulin (Jaulin, Kiefer, Didrit, & Walter, 2001) which is named SIVIA (Set Inverter Via Interval Analysis). It uses an interval Boolean function \([\text{test}](\mathcal{X})\) that if it gives as result 1, then \( \mathcal{X} \subseteq \mathcal{X} \), if it gives as result 0, then \( \mathcal{X} \cap \mathcal{X} = \emptyset \), and it gives \([0,1]\) as result if no conclusion could be made. The SIVIA algorithm in pseudo code is as follows:

**Algorithm SIVIA**(in: \( \text{test} \), \( \mathcal{X} \), \( \epsilon \); inout: \( \overline{\mathcal{X}} \), \( \bar{\mathcal{X}} \))

1. if \([\text{test}](\mathcal{X}) = 0\) return;
2. if \([\text{test}](\mathcal{X}) = 1\) \{\( \overline{\mathcal{X}} := \overline{\mathcal{X}} \cup \mathcal{X} \); \( \bar{\mathcal{X}} := \bar{\mathcal{X}} \cup \mathcal{X} \); return\};
3. if \(w((\mathcal{X})) < \epsilon\) \{\( \overline{\mathcal{X}} := \overline{\mathcal{X}} \cup \mathcal{X} \); return\};
4. SIVIA\(([\text{test}, L] [\mathcal{X}], \epsilon, \overline{\mathcal{X}}, \bar{\mathcal{X}})\); SIVIA\(([\text{test}, R] [\mathcal{X}], \epsilon, \overline{\mathcal{X}}, \bar{\mathcal{X}})\);

In SIVIA a natural choice for initializing \( \overline{\mathcal{X}} \) and \( \bar{\mathcal{X}} \) would be \( \overline{\mathcal{X}} = \bar{\mathcal{X}} = \emptyset \). The choice of \( \epsilon \) controls the precision of the enclosure, since the smallest boxes \( \mathcal{X} \) on the subpaving are the first boxes, obtained by successive bisections, for which \( w((\mathcal{X})) < \epsilon \). A smaller \( \epsilon \) yields a tighter enclosure but increases the computational time.

A possible way of improving the over-approximation is the use of contractors instead of test functions.

**Definition III**

**Contractors** are functions \( \mathcal{C}_\mathcal{S} : \mathbb{I} \mathbb{R}^n \to \mathbb{I} \mathbb{R}^n \) from interval vectors to interval vectors that satisfy the following conditions (\( \mathcal{S} \) is a set associated with the contractor):

\[
\forall [\mathcal{X}], \mathcal{C}_\mathcal{S}([\mathcal{X}]) \subseteq [\mathcal{X}];
\]

(\textit{contractance}),

\[
\forall [\mathcal{X}], [\mathcal{X}] \cap \mathcal{S} \subseteq \mathcal{C}_\mathcal{S}([\mathcal{X}]);
\]

(\textit{correctness}).

The SIVIA algorithm can be adapted for use with contractors, yielding a non-regular subpaving. The algorithm is as follows:

**Algorithm SIVIA**(in: \( \mathcal{C}_\mathcal{X}, [\mathcal{X}], \epsilon \); inout: \( \overline{\mathcal{X}} \))

1. \( [\mathcal{X}] := \mathcal{C}_\mathcal{X}([\mathcal{X}]) \);
2. if \( [\mathcal{X}] = \emptyset \) return;
3. if \( w((\mathcal{X})) < \epsilon \) \{\( \overline{\mathcal{X}} := \overline{\mathcal{X}} \cup [\mathcal{X}] \); return\};
4. SIVIA\(([\mathcal{C}_\mathcal{X}, L] [\mathcal{X}], \epsilon, \overline{\mathcal{X}})\); SIVIA\(([\mathcal{C}_\mathcal{X}, R] [\mathcal{X}], \epsilon, \overline{\mathcal{X}})\);
For the under-approximation we can use another type of functions from interval vectors to interval vectors $\mathcal{F}: \mathbb{I}^n \to \mathbb{I}^n$. On this work they will be named fitters. For fitters the correctness condition changes:

\begin{align*}
\forall [x], \mathcal{F}_x([x]) \subseteq [x] & \quad \text{(contractance),} \\
\forall [x], \mathcal{F}_x([x]) \subseteq [x] \cap \mathcal{S} & \quad \text{(correctness),}
\end{align*}

The SIVIA algorithm for use with fitters is described below.

**Algorithm** SIVIA(in: $\mathcal{F}_x$, $[x]$, $\epsilon$; inout: $\mathcal{X}$)

1. $[x_f] := \mathcal{F}_x([x])$ ;
2. if $[x_f] = \emptyset$ return;
3. if $w([x_f]) < \epsilon$ ($\mathcal{X} := \mathcal{X} \cup [x_f]$; return);;
4. SIVIA($\mathcal{F}_x$, $L[x]$, $\epsilon$, $\mathcal{X}$); SIVIA($\mathcal{F}_x$, $R[x]$, $\epsilon$, $\mathcal{X}$);

As for the use of contractors, SIVIA with fitters yields a non-regular subpaving.

It is important to note that the computation of an inner bound can be performed in the same way as an outer bound and the opposite is also true. We denote the complementary set of $\mathcal{X}$ as follows:

$$\mathcal{X}^c = \mathbb{R}^n \setminus \mathcal{X} \quad (3-57)$$

As properties of the complementary set we have that:

$$\mathcal{X}^{cc} = \mathcal{X} \quad (3-58)$$

$$\mathcal{X} \subseteq \mathcal{Y} \iff \mathcal{X}^c \supseteq \mathcal{Y}^c \quad (3-59)$$

We can compute an inner approximation of the set $\mathcal{X}$ using some method of computing an outer approximation by considering $\mathcal{X}^c = \overline{\mathcal{X}^c}$. $\overline{\mathcal{X}^c}$ is an under approximation of $\mathcal{X}$ since $\overline{\mathcal{X}^c} \supseteq \mathcal{X}^c$, by (3-59) we have that $\overline{\mathcal{X}^c} \subseteq \mathcal{X}^c$ and using (3-57) we have that $\overline{\mathcal{X}^c} \subseteq \mathcal{X}$.

To study the overestimation of a subpaving $\overline{\mathcal{X}}$ we may denote the list of boxes on the subpaving as $\overline{\mathcal{X}}$ and bound the overestimation, $\text{oe} = \max_{x \in \overline{\mathcal{X}}} \min_{y \in \mathcal{X}} \|x - y\|$ by (3-60).

$$\text{oe} \leq \max_{[x] \in \mathcal{X}} \left( \max_{y \in [x]} \|x - y\| + \min_{x \in [x], y \in \mathcal{X}} \|x - y\| \right) \leq \max_{x,y \in [x], y \in \mathcal{X}} \|x - y\| + \max_{x \in \mathcal{X}} \min_{y \in \mathcal{X}} \|x - y\| \quad (3-60)$$

In the previous expression the norm $\|x\|$ is defined as follows.

$$\|x\| = \sqrt{\sum_{i=1}^{n} x_i^2}$$
Denoting \( W = \{ [x] \in \mathbb{R}^n : w([x]) \leq \varepsilon, [test](x) = 1 \} \) or, if using contractors, \( W = \{ [x] \in \mathbb{R}^n : w([x]) \leq \varepsilon, C_X([x]) \neq \emptyset \} \) we have the following relations.

\[
\max_{[x] \in W} \left( \min_{y \in [x]} \| x - y \| \right) \leq \sqrt{n} \varepsilon \leq oe_{sp}(\varepsilon)
\]

When using a test function we have the following relation.

\[
\max_{[x] \in W} \left( \min_{y \in [x]} \| x - y \| \right) \leq \max_{[x] \in W} \left( \min_{y \in [x]} \| x - y \| \right) \leq oe_{test}(\varepsilon, test)
\]

And when using contractors we have the following equivalent definition.

\[
\max_{[x] \in W} \left( \min_{y \in [x]} \| x - y \| \right) \leq oe_{ct}(\varepsilon, C_X)
\]

when using contractors. Then the overestimation can be bounded using (3-61).

\[
oe \leq oe_{sp}(\varepsilon) + oe_{test}(\varepsilon, test) \text{ or } oe \leq oe_{sp}(\varepsilon) + oe_{ct}(\varepsilon, C_X)
\] (3-61)

From (3-60) we can see that the overestimation can be bounded by a term that depends only on \( \varepsilon \), and a term that depends on \( \varepsilon \) and the test \( test \) or the contractor \( C_X \). The overestimation \( oe \) can tend to 0 only if \( \varepsilon \) tends to 0 and if \( test \) or \( C_X \) are improved, that is, if they yield tighter results.

3.7 Conclusions

In this chapter we have seen that the field of interval analysis allows us to deal with interval numbers extending many of the operations on real numbers to be defined on interval numbers.

There are many ways of defining an inclusion function to make real functions also admit interval inputs. Among those there are the interval extension, the centered form, the mean value extension, the mixed centered inclusion function, the slope form and the Taylor expansion. Each form has its advantages and disadvantages depending on the function to be enclosed.

Also in this chapter the integration of functions from real numbers to intervals was defined and a method of computing an enclosure of the solution (guaranteed integration) based on the Picard iterates was described.

We have also seen that through Interval Taylor Models we are able to perform a better interval bounding of a function than by partitioning the domain and evaluating the interval extension of a function. Also, we have seen that Interval Taylor Models can also be considered as wrappers for sets. Another interest of using ITMs is that it is possible to define operations on ITMs yielding the Remainder Differential Algebra (RDA).

In this chapter it was also observed that there is much work done on solvers of IVPs that perform guaranteed integration.
Finally, it was seen that a better way of wrapping a set, instead of enclosing it on an interval vector, is to enclose it on an union of non-overlapping boxes, a subpaving. A method of computing subpavings that contain a set is set inversion namely by the SIVIA algorithm, which uses test functions or contractors to determine if a box does not intersect with the set or which part of the box may intersect the set.

Chapter 4: The Proposed Solution

In this work the proposed solution is computing subpavings containing or included on the reachable sets at different time steps. A similar theory for discrete systems can be found in the book of Luc Jaulin (Jaulin, Kiefer, Didrit, & Walter, 2001). For continuous systems, the problem of bounding reachable sets independently of time has been addressed in the paper (Lhommeau, Jaulin, & Hardouin, 2011). However, the problem of bounding reachable sets for a specific finite time through subpavings has not been discussed yet. The problem of bounding reachable sets for a specific finite time is discussed in this chapter, and the formulation necessary to solve that problem is presented.

In the first section of this chapter, 4.1, the theory and an application example in (Lhommeau, Jaulin, & Hardouin, 2011) are summarized and discussed. In section 4.2, the theory developed in this work for the problem at hand is presented. Section 4.3 presents the main conclusions from this chapter.

4.1 Theoretical background - Method of Jaulin

The method of Jaulin to compute guaranteed bounds of reachable sets is described in the paper (Lhommeau, Jaulin, & Hardouin, 2011) submitted in 2007 that was written by Mehdi Lhommeau and Laurent Hardouin from the Laboratoire d'Ingénierie des Systèmes Automatisés, Université d'Angers, and by Luc Jaulin from ENSIETA.

This work aims to compute guaranteed bounds on the capture basin of a dynamical system $C$ defined as follows:

**Definition III** (Time independent capture basin). The time independent capture basin $C$ is the set of all states $x$ such that, for some control $u(t) \in \mathcal{U}(0 \leq t \leq t_f)$, there exists a trajectory that can reach the target set $T$, that is, $\varphi(t_f, x, u) \in T$ without leaving the allowable set of states $\mathcal{K}$ in a finite time, that is, $\varphi([0, t_f], x, u) \subset \mathcal{K}$.

On the previous definition we used the extension of $\varphi(t, x_0, u)$ to admit interval arguments $\varphi([t], [x], [u])$ which is defined as:

$$\varphi([t], [x], [u]) \triangleq \{x \in \mathbb{R}^n : \exists t \in [t], \exists x_0 \in [x], \exists u \in [u], x = \varphi(t, x_0, u)\}.$$

The algorithm developed in that work computes an inclusion function of the trajectory $\varphi([t], [x], u)$ that determines the states reached in a time $t$ for initial conditions in the box $[x]$. This inclusion of the flow
can either be done by simple Euler integration or by more complex verified integration programs, such as VSPODE or VNODE. In the work of Jaulin, Euler integration was used. The flow is evaluated in a box $[w]$, that includes $[x]$, which is an expansion of $[v] = [x](t_1) \cup [x](t_2)$ as is shown in Figure 4-1, where $[x](t_2) = [x](t_1) + (t_2 - t_1)[f](x(t_1), u)$. Then we compute $[x](t_2) = [x](t_1) + (t_2 - t_1)[f]([w], u)$.

If $[x](t_2) \subset [w]$ the result is valid and $[\varphi](t_2 - t_1, [x](t_1), u) = [x](t_2)$, otherwise it was not possible to reach a conclusion and we consider $[\varphi](t_2 - t_1, [x](t_1), u) = \mathbb{R}^n$. The box $[w]$ is computed with the function $[w] = inflate([v], \alpha \ast w([v]) + \beta)$, where the function $inflate([x], \epsilon)$ is defined as follows:

$$
inflate([x], \epsilon) = ([\inf([x_1]) - \epsilon, \sup([x_1]) + \epsilon], ..., [\inf([x_n]) - \epsilon, \sup([x_n]) + \epsilon]) (4-1)
$$

Figure 4-1: Figure illustrating the principle of $[\varphi]$

To compute guaranteed bounds on the capture basin the algorithm uses a branch and prune procedure based on theorem V.

**Theorem V**

If $\mathcal{C}^-$ and $\mathcal{C}^+$ are such that $\mathcal{C}^- \subset \mathcal{C} \subset \mathcal{C}^+ \subset \mathbb{R}$, if $[x]$ is a box and if $u \in \mathcal{F}([0, t] \rightarrow \mathbb{U})$, then

- $[x] \subset T \Rightarrow [x] \subset \mathcal{C}$ (4-2)
- $[x] \cap \mathbb{R} = \emptyset \Rightarrow [x] \cap \mathcal{C} = \emptyset$ (4-3)
- $(\varphi(t, [x], u) \subset \mathcal{C}^- \land \varphi([0, t], [x], u) \subset \mathbb{R}) \Rightarrow [x] \subset \mathcal{C}$ (4-4)
- $\varphi(t, [x], \mathbb{U}) \cap \mathcal{C}^+ = \emptyset \land \varphi(t, [x], \mathbb{U}) \cap \mathbb{R} = \emptyset \Rightarrow [x] \cap \mathcal{C} = \emptyset$ (4-5)

Figure 4-2: Illustration of theorem V

The branch and prune procedure tests boxes based on the previous theorem, and then take the boxes where we cannot take any conclusion and divides them into smaller boxes that will be tested, as in the SIVIA algorithm. The result on a practical example of the Zermelo’s problem is shown in Figure 4-3.

The dynamics of the Zermelo’s problem is given by the following differential equations.
The two control inputs are the linear velocity module \(0 \leq v \leq 0.8\) and the linear velocity angle \(\theta \in [-\pi, \pi]\). The target set considered is the circle with center at \((0, 0)\) and radius 1. The dark gray area represents \(C^-\) and the light gray area represents \(C^+ \setminus C^-\).

\[
\begin{align*}
\dot{x}_1(t) &= 1 + \frac{x_2^2 - x_1^2}{(x_1^2 + x_2^2)^2} + v \cos(\theta) \\
\dot{x}_2(t) &= \frac{-2x_1x_2}{(x_1^2 + x_2^2)^2} + v \sin(\theta)
\end{align*}
\] (4-6) (4-7)

The developed algorithm has the advantage that, since it uses interval analysis, it gives guaranteed bounds on the reachable set, and since it uses a branch and prune procedure, we can have a prescribed accuracy. On the downside we have that the branch and prune procedure causes the algorithm to require a high computational effort for high dimensions or small target sets, and that the described inclusion function of the trajectory uses values of the flow field in the set of initial conditions, which is not necessary.

### 4.2 Formulation for Reachable sets

Before describing a formulation that can be used to compute enclosures of reachable sets, a useful assumption (Assumption I) that allows the use of methods to compute outer approximations of reachable sets to compute inner approximations is needed first.

**Assumption I**

If \(\mathcal{T}(\tau)\) is the backward reachable set at time \(\tau\), of the system (2-1) from the target set \(\mathcal{T}_0\), then \(\mathcal{T}^- (\tau)\) is the backward reachable set at time \(\tau\), of the system (4-8) from the target set \(\mathcal{T}_0^-\):

\[
\begin{align*}
\dot{x} &= f(x(t), d(t), u(t)) \\
x(t_f) &\in \mathcal{T}_0^-, t \in [0, t_f]
\end{align*}
\] (4-8)

where \(0 \leq t_f < \infty, x \in \mathbb{R}^n\) is the state, \(u \in \mathbb{R}^p\) is the control input, \(d \in \mathbb{R}^m\) is the disturbance input. The spaces of admissible control input and disturbance input trajectories are denoted as the spaces of piecewise continuous functions \(\mathcal{U} = \{u(.) \in PC^n | u(t) \in \mathbb{R}, 0 \leq t \leq t_f\}\), and \(\mathcal{D} = \{d(.) \in PC^m | d(t) \in \mathbb{R}, 0 \leq t \leq t_f\}\) respectively.
That is, $\mathcal{T}^c(\tau)$ is the backward reachable set at time $\tau$, from the target set $\mathcal{T}_0^c$ treating the disturbance input of (2-1) as a control input and the control input of (2-1) as a disturbance input.

**Sketch of Proof of Assumption I**

To prove Assumption I we must prove that if $x \notin \mathcal{T}(\tau)$ then $x$ belongs to the backwards reachable set of the system (4-8) from the target set $\mathcal{T}_0^c$.

From definition I.2, if $x \notin \mathcal{T}(\tau)$ for every strategy $u(t) = \gamma[d](t) \in \mathcal{U}(T \leq t \leq t_f)$ there is a disturbance input $d(t) \in \mathcal{D}(T \leq t \leq t_f)$ such that $x(t_f) \notin \mathcal{T}_0$.

It may be possible to construct a strategy $\beta: \mathcal{U} \rightarrow \mathcal{D}$ by considering a time $r$ starting at $\tau$ and going towards $t_f$.

To each $u_i(t) \in \mathcal{U}(T \leq t \leq r)$, if $\beta[u](t)$ is not already defined for some $u(t)$ where $u(t) = u_i(t)$ for $T \leq t \leq r$ we have to define $\beta[u](t)$. Because we are considering $r$ starting at $\tau$ and going towards $t_f$, or $r = \tau$ or there is a function $u_d(t) = u_i(t)$ for $T \leq t < r$ where $\beta[u_d](t)$ is defined, on the last case we can consider $d_i(t) = \beta[u_d](t)$.

Then we take a strategy $\gamma: \mathcal{D}(T \leq t \leq T_f) \rightarrow \mathcal{U}(T \leq t \leq t_f)$ such that $\gamma[d](r) = u_i(r)$ for all $d(t) \in \mathcal{D}(T \leq t \leq t_f)$, and we take the disturbance input $d_i(t) \in \mathcal{D}(T \leq t \leq t_f)$ such that $x(t_f) \notin \mathcal{T}_0$ for that strategy.

We define for $u(t) = \{u_i(t) \text{ for } T \leq t < r, u_i(t) \text{ for } r \leq T \leq t_f\}$, $\beta[u](t)$ is defined as:

$$\beta[u](t) = \begin{cases} d_i(t) \text{ for } T \leq t < r \\ d_r(t) \text{ for } r \leq T \leq t_f \end{cases}$$

The strategy $\beta: \mathcal{U} \rightarrow \mathcal{D}$ should be a non-anticipative strategy $d(t) = \beta[u](t) \in \mathcal{D}(T \leq t \leq t_f)$ such that for every control input $x(t_f) \notin \mathcal{T}_0$, which is equivalent to $x(t_f) \in \mathcal{T}_0^c$.

Assumption I combined with the result $\mathcal{X}^c \subseteq \mathcal{X}$ shows that the computation of inner approximations of reachable sets can be transformed to outer approximations of reachable sets.

A similar procedure from that of the paper of Jaulin (Lhommeau, Jaulin, & Hardouin, 2011) could be used to compute enclosures for reachable sets and is based on Assumption II.

**Assumption II**

If $\mathcal{T}^-\tau(t)$ and $\mathcal{T}^+(t)$ are such that $\mathcal{T}^-\tau(t) \subseteq \mathcal{T}(t) \subseteq \mathcal{T}^+(t)$, if $[x] \in \mathbb{R}^n$ is a box of states, if $u \in \mathcal{U}$ and if $d \in \mathcal{D}$ then

- $[\varphi](\tau, [x], u, [\mathcal{D}]) \subseteq \mathcal{T}^-\tau(t) \Rightarrow [x] \subseteq \mathcal{T}(t - \tau)$ (4-8)
- $[\varphi](\tau, [x], [\mathcal{U}], d) \cap \mathcal{T}^+(t) = \emptyset \Rightarrow [x] \cap \mathcal{T}(t - \tau) = \emptyset$ (4-9)
Sketch of Proof of Assumption I

Since \( \phi(t,[x],U,d) \subseteq [\phi](t,[x],U,d) \) and \( T(t) \subseteq \mathcal{T}^+(t) \), \( [\phi](t,[x],[U],d) \cap \mathcal{T}^+(t) = \emptyset \Rightarrow \phi(t,[x],U,d) \cap T(t) = \emptyset \). From definition I.2 there is at least one disturbance input that, for all admissible control inputs, \( x(t) \notin T(t) \) from all states \( x(t-t) \in [x] \), and thus for all states \( x(t-t) \in [x] \), for every strategy \( \gamma : \mathcal{D} \rightarrow \mathcal{U} \) there is at least one disturbance input \( d(t) \) such that for a control input \( \gamma[d](t), x(t_f) \notin \mathcal{T}_0 \). This proves (4-9).

Noting that \( [x] \subseteq T(t-t) \Rightarrow [x] \cap \mathcal{T}^c(t-t) = \emptyset \), and that \( [\phi](t,[x],u,[D]) \subseteq \mathcal{T}^-(t) \Rightarrow \phi(t,[x],u,D) \subseteq T(t) \Leftrightarrow \phi(t,[x],u,D) \cap \mathcal{T}^c(t) = \emptyset \) we can say that (4-8) follows from Assumption I and (4-9). ■

Figure 4-4: Illustration of assumption II

In Assumption II we consider a trajectory of the type \( \phi(t,x_0,u,d) \) which is the solution \( x(t) \) of the initial value problem \( \dot{x} = f(x,u,d) \) for the initial vector \( x_0 \), that is, the input of the system function is divided in control input \( u \), and disturbance input \( d \). To admit interval arguments we define \( \phi([t],[x],[u],[d]) \equiv \{x \in \mathbb{R}^n : \exists \tau \in [t] \exists x_0 \in [x], \exists u \in [u], \exists d \in [d], x = \phi(t,x_0,u,d) \} \).

To get tighter bounds and to improve convergence, it is better to use contractors (definition III) instead of test functions. For that purpose we can use Assumption III. In Assumption III we consider an inverse trajectory \( \phi_{inv}(t_f,x_0,u,d) \) that, if \( \phi(t_f,x_0,u,d) \) is the solution of \( \dot{x} = f(x,u,d) \), \( \phi_{inv}(t_f,x_0,u,d) \) is the solution of \( \dot{x} = -f(x,u_{inv},d_{inv}) \), where \( u_{inv}(t) = u(t_f-t) \) and \( d_{inv}(t) = d(t_f-t) \).

Assumption III

If \( \mathcal{T}^- \) and \( \mathcal{T}^+ \) are such that \( \mathcal{T}^- \subseteq T(t) \subseteq \mathcal{T}^+ \), if \( [x] \in \mathbb{R}^n \) is a box of states, if \( u \in \mathcal{U} \) and if \( d \in \mathcal{D} \) then

- The interval function \( C_{\mathcal{T}^+(t-t)}([x]) = [x] \cap [\phi_{inv}](t,[\phi](t,[x],u,[D]) \cap \mathcal{T}^-(t),u,[D]) \) (4-10) is a contractor for the constraint \( x \in \mathcal{T}^+(t-t) \)

- The interval function \( C_{\mathcal{T}^-(t-t)}([x]) = [x] \cap [\phi_{inv}](t,[\phi](t,[x],u,d) \cap \mathcal{T}^+(t),[u],d) \) (4-11) is a contractor for the constraint \( x \in \mathcal{T}(t-t) \)

34
Sketch of Proof of Assumption III

The contractance of (4-11) and (4-10) is proven by the fact that for any \([x], [y] \in \mathbb{R}^n \), \([x] \cap [y] \subseteq [x] \).

\[\varphi([r],[x],[\mathcal{U}],d) \subseteq \varphi(r,[x],[\mathcal{U}],d) \cap T(t) \]

is an enclosure of the states on \(T(t)\) and each system can reach from the original box \([x]\) by using as control some strategy \(\gamma : \mathcal{D} \rightarrow \mathcal{U}\) and as disturbance input \(d\).

From definition I.2, we have that \(\varphi([r],[x],[\mathcal{U}],d) \times T^+(t) \]

and thus \(\varphi_{\text{inv}}([r],[x],[\mathcal{U}],d) \subseteq \varphi_{\text{inv}}([r],[x],[\mathcal{U}],d) \times T^+(t) \) proving the correctness of (4-11).

(4-10) follows directly from Assumption I and (4-11).

\[\text{Figure 4-5: Illustration of assumption III}\]

From Assumptions II and III, by starting with a box \([x_0] \rightarrow T(t + \tau)\) and having a description of \(T^+(t)\) and \(T^-(t)\), using a branch and prune algorithm we can obtain an inner or an outer bound for the reachable set \(T(t - \tau)\).

To obtain a tighter enclosure, given the target set \(T_0\) we can compute inner and outer bounds for \(T(0)\) using a number of time steps \(N_{\text{stp}}\) and using \(T^-(t_f - i \times t_f / N_{\text{stp}})\) and \(T^+(t_f - i \times t_f / N_{\text{stp}})\) as intermediate results, where \(i\) is an integer from 0 to \(N_{\text{stp}} - 1\). Using intermediate time steps can help reduce the effects associated with the overestimation of \(\varphi([r],[x],[\mathcal{U}],d)\), because it increases nonlinearly with \(\tau\). On the other hand, with more time steps the accumulated error increases since on each time step we are overestimating the reachable set at most by the size of the smallest box on the subpaving plus the highest overestimation of \(\varphi([r],[x],[\mathcal{U}],d)\) and the error associated with assumption II or III, as will be seen.

For a system without disturbance, if we are describing \(T^+(t)\) by a subpaving obtained by a branch and prune or a branch and contract algorithm \(N_{\text{stp}}\) time steps, we denote \(o\varepsilon_i\) as the overestimation at the \(i\)th time step. The overestimation \(o\varepsilon_i\) can be bounded using equation (3-60) here applied for the \(i\)th time step.

\[o\varepsilon_i \leq o\varepsilon_{\text{sp}}(\varepsilon) + o\varepsilon_{\text{test}}(\varepsilon,\text{test}) \quad \text{or} \quad o\varepsilon_i \leq o\varepsilon_{\text{sp}}(\varepsilon) + o\varepsilon_{\text{ct}}(\varepsilon, C_{T(t_i-t/N_{\text{stp}})}) \quad (4-12)\]
The overestimation of a test function or contractor based on assumptions II or III, \( o_{e_{\text{test,II}}} \) or \( o_{e_{\text{ct,II}}} \), can be limited by the overestimation caused by the overestimation on the previous time step, \( o_{e_{\text{prev}}} \), plus the overestimation caused by the inclusion function of the trajectory \([\varphi] \), \( o_{e_{\text{enclose}}} \), plus the overestimation assumed with assumptions II or III, \( o_{e_{\text{assump}}} \).

Considering \( \mathcal{O} \) as the set of samples of \( \mathcal{D} \) used as disturbance control on the test function or contractor, and considering the set \( \mathcal{H} \equiv \{x \in \mathbb{R}^n : [x] \in \mathcal{T}^+(0), w([x]) \leq \varepsilon \} \) we define the growth factor (the maximum expansion factor between two points under this method for a time step) as follows:

\[
gf(t) \equiv \max_{x,y \in \mathcal{H}, d \in \mathcal{D}, u \in \mathcal{U}} \left( \frac{\|x-y\|}{\|\varphi(t,x,u,d) - \varphi(t,y,u,d)\|} \right) \tag{4-13}
\]

Denoting \( S(t,[x]) \) as the forward reachable set at time \( t \) of the system (2-1) from the initial set \( S_0 = [x] \), we can define the overestimation due to the assumption II or III, \( o_{e_{\text{assump}}} \), by (4-14).

\[
oe_{\text{assump}}(t, \mathcal{O}, ...) \equiv \max_{x \in \mathcal{H}, y \in \mathcal{H}, d \in \mathcal{D}, u \in \mathcal{U}} \min_{y \in \mathcal{S}(t,[x])} \|x-y\| \tag{4-14}
\]

A minimal inclusion test for \( \mathcal{T}(t - \tau) \) is based on the equivalence \( [x] \cap \mathcal{T}(t - \tau) \neq \emptyset \iff S(t,[x]) \cap \mathcal{T}(t) \neq \emptyset \). \( o_{e_{\text{assump}}}(t, \mathcal{O}, ...) \) is the maximal error on \( \mathcal{T}(t) \) by approximating \( S(t,[x]) \) by \( \bigcap_{d \in \mathcal{D}} \varphi(t,[x],u,d) \supset S(t,[x]) \). In the frequent case that the optimal disturbance input to maximize the time to reach is of type "bang-bang", and the set of samples \( \mathcal{O} \) contains only the cases where the disturbance input is constant and equal to the lower or upper bound, a shorter time step may reduce this overestimation since, if the switching time on the optimal disturbance input occurs during that time step, the enclosed trajectory is non-optimal for a shorter time.

We may also compute bounds on the overestimation caused by the method of enclosing the trajectory \([\varphi](t,[x],\mathcal{U},d), o_{e_{\text{enclose}}} \). Such bound is given by equation (4-15).

\[
oe_{\text{enclose}}(t, [\varphi], ...) \equiv \max_{x \in \mathcal{H}, y \in \mathcal{H}, d \in \mathcal{D} \cap \mathcal{E}(\varphi)(t,[x],\mathcal{U},d)} \min_{y \in \mathcal{E}(\varphi(t,[x],u,d))} \|x-y\| \tag{4-15}
\]

The sources of the overestimation \( o_{e_{\text{enclose}}} \) are the common sources of overestimation on interval analysis, such as the dependency effect and the wrapping effect. Since the inclusion function of the trajectory is implemented on a computer, on \( o_{e_{\text{enclose}}} \) the numerical overestimation associated with a finite precision arithmetic is also considered. Typically, in non-linear problems, since the dependency effect increases with the size of the interval argument, \( o_{e_{\text{enclose}}} \) grows faster than linearly with time \( t \).

Also, from the dependency effect and the wrapping effect it follows that some of this overestimation \( o_{e_{\text{enclose}}} \) is unavoidable for some problems. If every component of the disturbance input \( d \) appears only once in the expression of \( f(x,\mathcal{U},d) \) it follows that theoretically in a large number of cases, if \( \mathcal{U} \) is a box \( f(x,\mathcal{U},d) = [f](x,\mathcal{U},d) \). Otherwise, if there are multiple occurrences of any component of \( u \) in the
expression of \( f(x,u,d) \) there can be an unavoidable overestimation of \( f(x,u,d) \) when enclosing it by \([f](x,\mathbb{U},d)\) and thus independently of the integration method used there is an unavoidable overestimation of \([\varphi](t,[x],\mathbb{U},d)\) and therefore also of \( \text{oe}_{\text{enclose}} \).

The overestimation due to considering the target set, at the \( i \)th time step, as \( T^{+}(t_{f} - i \ast t_{f}/\text{Nstp}) \), instead of \( T(t_{f} - i \ast t_{f}/\text{Nstp}) \), is limited by \( \text{oe}_{\text{prev}} \) defined on (4-16).

\[
\text{oe}_{\text{prev}} \equiv gf(t)\text{oe}_{i-1} \tag{4-16}
\]

As a result from (4-14), (4-15) and (4-16) we have that:

\[
\text{oe}_{\text{test}}(\varepsilon,t) \leq \text{oe}_{\text{assump}}(t/\text{Nstp},\mathbb{O},...) + \text{oe}_{\text{enclose}}(t/\text{Nstp},[\varphi],...) + \text{oe}_{\text{prev}} \tag{4-17}
\]

Or

\[
\text{oe}_{\text{ct}}(\varepsilon,C_{T(t-i-t/\text{Nstp})}) \leq \text{oe}_{\text{assump}}(t/\text{Nstp},\mathbb{O},...) + \text{oe}_{\text{enclose}}(t/\text{Nstp},[\varphi],...) + \text{oe}_{\text{prev}} \tag{4-18}
\]

Combining (4-12) with (4-17) or (4-18) and (4-16) we have:

\[
\text{oe}_{i} \leq \text{oe}_{sp}(\varepsilon) + \text{oe}_{\text{assump}}(t/\text{Nstp},\mathbb{O},...) + \text{oe}_{\text{enclose}}(t/\text{Nstp},[\varphi],...) + gf(t/\text{Nstp})\text{oe}_{i-1} \tag{4-19}
\]

We define the overestimation of a time step as \( \text{oe}_{\text{tstep}} \):

\[
\text{oe}_{\text{tstep}} \equiv \text{oe}_{sp}(\varepsilon) + \text{oe}_{\text{assump}}(t/\text{Nstp},\mathbb{O},...) + \text{oe}_{\text{enclose}}(t/\text{Nstp},[\varphi],...) \tag{4-20}
\]

By recursion we have equation (4-21).

\[
\text{oe}_{\text{Nstp}} \leq \text{oe}_{\text{tstep}} + \text{oe}_{\text{tstep}}gf(t/\text{Nstp}) + \cdots + \text{oe}_{\text{tstep}}gf(t/\text{Nstp})^{\text{Nstp}-1} = \begin{cases} 
\text{oe}_{\text{tstep}}\left(\frac{1-gf(t/\text{Nstp})^{\text{Nstp}}}{1-gf(t/\text{Nstp})}\right), & gf(t/\text{Nstp}) \neq 1 \\
\text{oe}_{\text{tstep}}N\text{stp}, & gf(t/\text{Nstp}) = 1 
\end{cases} \tag{4-21}
\]

For a large number of steps \( \text{Nstp} \), \( gf(t/\text{Nstp}) \approx 1 \), thus, for a large number of steps the overestimation is roughly bounded by \( \text{oe}_{\text{Nstp}} \) (4-22).

\[
\text{oe}_{\text{Nstp}} \equiv \text{oe}_{sp}(\varepsilon)\text{Nstp} + \text{oe}_{\text{assump}}(t/\text{Nstp},\mathbb{O},...)\text{Nstp} + \text{oe}_{\text{enclose}}(t/\text{Nstp},[\varphi],...)\text{Nstp} \tag{4-22}
\]

The overestimation due to approximating the reachable set by a subpaving is of the order of \( \text{oe}_{sp}(\varepsilon)\text{Nstp} \) and grows linearly with \( \text{Nstp} \). The overestimation due to the overestimation of the inclusion function of the flow \([\varphi], \text{oe}_{\text{enclose}}(t/\text{Nstp},[\varphi],...)\text{Nstp} \), decreases with \( \text{Nstp} \) because \( \text{oe}_{\text{enclose}}(t,[\varphi],...) \) increases faster than linearly with \( \tau \). From this analysis it can be concluded that there must exist an optimal number of steps \( \text{Nstp} \) and that number is approximately given by

\[
\frac{\partial}{\partial \tau}(\text{oe}_{\text{enclose}}(t,[\varphi],...)) \bigg|_{\text{Nstp}} \approx \frac{\tau}{\text{Nstp}} \approx \text{oe}_{sp}(\varepsilon) = \sqrt{\text{Nstp}}. \quad \text{That condition can be formulated as}
\]

37
\[ C \equiv \frac{1}{\sqrt{h}} \frac{\partial}{\partial \tau} (\alpha e_{\text{enclose}}(\tau, [\varphi], \ldots)) \frac{\tau / \text{Nstp}}{\varepsilon} \approx 1. \] This condition resembles the Courant-Friedrichs-Lewy condition for finite difference methods, since \( t / \text{Nstp} \) can be regarded as the time step and \( \varepsilon \) as a grid size.

### 4.3 Conclusions

From the first section of this chapter we conclude that there is already a method of computing guaranteed bounds of the time independent capture basin, of definition III, through subpavings, however that method differs from what is aimed in this work because the time independent capture basin is equivalent to a reachable set when the final time tends to infinity and here we aim to compute bounds when the final time is finite.

From section 4.2 we can conclude that assumptions II and III provide the necessary formulations for a method of determining guaranteed bounds on reachable sets through subpavings.

Finally we have seen that the overestimation of the method formulated in section 4.2 for a given \( \varepsilon \) and assuming that there is no overestimation due to the sampling of the control should have a minimum value when the number of time steps \( \text{Nstp} \) provides a balance between the overestimation due to wrapping the intermediate results with subpavings and the overestimation of the method of computing the guaranteed integration of ODEs. Such a number is reached when the condition

\[ C \equiv \frac{1}{\sqrt{h}} \frac{\partial}{\partial \tau} (\alpha e_{\text{enclose}}(\tau, [\varphi], \ldots)) \frac{\tau / \text{Nstp}}{\varepsilon} \approx 1 \] is met.

## Chapter 5: Computational Application

This chapter holds the description of the computational application of the method described in the last chapter. This chapter is organized as follows. In section 5.1 an algorithm that performs a branch and prune procedure to generate subpavings is described. In the following section, 5.2, a modified version of the algorithm of the previous section, that is more efficient when the test function is invariant, is described. On section 5.3 a series of programming features that may be useful to improve computational efficiency are explained. The next section, 5.4, provides the description of an algorithm that uses contractors to compute subpavings. The following section, 5.5, explains how the previous algorithms can be applied to compute outer bounds of reachable sets. Section 5.6 describes the necessary alterations to compute inner bounds. After this section, section 5.7 describes the implementation of guaranteed integration using the Euler method. Section 5.8 describes the implementation of VSPODE. The last section, 5.9, contains the conclusions from this chapter.

### 5.1 Branch and prune (B&P) algorithm

A generic algorithm was developed to describe the outer approximation of a set \( \mathcal{T} \) through a subpaving \( \mathcal{S} \) of an initial box \([x_0]\). The elimination of branches is done by means of a test \( \text{test}([x]) \) that is false case a box does not belong to the set \( \mathcal{T} \), and true if no conclusion can be made. In this
algorithm, an outer approximation of the set $\mathcal{T}$ is considered as a parameter for the test which can be the subpaving $\mathcal{S}$ already obtained in the algorithm. If this is the case, it can happen that the result of the test may vary at different stages of the algorithm. This algorithm considers the subpaving $\mathcal{S}$ as a list of boxes. As inputs, the algorithm must take an initial over approximation of the set $\mathcal{T}$ as a box $[x_0]$ (which will be the root of the subpaving $\mathcal{S}$), the initial outer set of boxes in $\mathcal{E}$ and the parameters used in the test.

The algorithm is described on the flow chart of Figure 5-1. First the list $\mathcal{S}$ is initialized with $[x_0]$ and the list $\mathcal{E}$ is initialized with boxes in contact with $[x_0]$ and preferably not in contact with the set $\mathcal{T}$. We will symbolize the set containing those initial boxes as $\mathcal{J}$. List $\mathcal{E}$ will also contain the boxes eliminated from $\mathcal{S}$ during the algorithm. The purpose of having list $\mathcal{E}$ is to make sure that only boxes intersecting any box on that list are eliminated from $\mathcal{S}$ or bisected. This property could be useful because the boxes that do not intersect any box on list $\mathcal{E}$ likely belong to $\mathcal{T}$ and thus its evaluation or bisection should not be necessary. The initial number of boxes on $\mathcal{S}$ (NB) is one. The algorithm keeps refining the subpaving until every box on $\mathcal{S}$ intersecting a box on $\mathcal{E}$ has $w([x]) \leq \varepsilon$. This only happens (Check again is true) when, on the previous iteration, no box was eliminated from $\mathcal{S}$ or bisected. This is implemented by initializing ‘Check again’ as false and if any alteration on $\mathcal{S}$ occurs during the iteration ‘Check again’ becomes true.

On each iteration every box on $\mathcal{S}$ must be checked. This consists of a cycle that repeats itself for ‘initial NB’ times, which is the number of boxes on $\mathcal{S}$ (NB) at the beginning of the iteration. The cycle starts by removing the first box from $\mathcal{S}$ and attributing it to $[x]$. If the box $[x]$ does not intersect any box on $\mathcal{E}$, $[x]$ should stay on $\mathcal{S}$ and the box is inserted on the last entry of $\mathcal{S}$. If the box $[x]$ intersects any box on $\mathcal{E}$ it should be tested to verify if the box does not intersect the set $\mathcal{T}$. This is done by means of the Boolean function $\text{test}([x])$. If the evaluated box $[x]$ yields $\text{test}([x])$ false, the box does not intersect with the set $\mathcal{T}$ and therefore should not belong in the subpaving $\mathcal{S}$. If this is the case, box $[x]$ is not inserted back on $\mathcal{S}$ and is inserted as the last entry of list $\mathcal{E}$. On this case list $\mathcal{S}$ loses an element so NB should be subtracted by one and ‘Check again’ should become true. If $\text{test}([x])$ is true no conclusion can be made and $[x]$ should remain on the subpaving. If this is the case there are two possible outcomes. Or $w([x]) > \varepsilon$ and $[x]$ should be bisected, or $w([x]) \leq \varepsilon$ and it is not necessary to bisect $[x]$. On the first case $[x]$ is bisected and $L[x]$ and $R[x]$ are inserted on $\mathcal{S}$ instead of $[x]$. First $L[x]$ is inserted as the last entry of $\mathcal{S}$ and then $R[x]$ is inserted as the last entry of $\mathcal{S}$ after $L[x]$.

This order of inserting the boxes after the bisection results that the final list $\mathcal{S}$ is ordered as represented on the binary tree, with the branches more at left first on the list. In this case, a box is bisected on $\mathcal{S}$, then NB should be increased by one and ‘Check again’ becomes true. In the second case, box $[x]$ should remain on $\mathcal{S}$ then it is stored as the last entry on $\mathcal{S}$. Since in this case list $\mathcal{S}$ does not change, NB remains the same and ‘Check again’ remains the same.
As an example, to enclose the circle defined as $(x_1 - 0.5)^2 + (x_2 - 0.5)^2 < 2.25$ we may use the following test function:

$$test([x]) = \begin{cases} 
  \text{false}, & (|x_1| - 0.5)^2 + (|x_2| - 0.5)^2 \geq 2.25 \\
  \text{true}, & \text{otherwise} 
\end{cases} \quad (5-1)$$

As input parameters, we can use the initial box $[x]_0 = ((-2,2),(-2,2))$, an initial list $\mathcal{E} = \{((-2,2),(-2,2)), (2,(-2,2)), ((-2,2),-2), ((-2,2),2)\}$ and $\varepsilon = 1.1$. The result is drawn in Figure 5-2. The shaded areas are the boxes belonging to $\mathcal{S}$ at a given iteration (in light gray are the boxes where $test([x])$ is false and in dark gray the boxes where $test([x])$ is true) and the white boxes belong to the list $\mathcal{E}$. The circle in solid line represent the boundary of the set $\mathcal{T}$.

Figure 5-2: Evolution of lists $\mathcal{S}$ and $\mathcal{E}$
Table 5-1: List $\mathcal{E}$ after branch-and-prune algorithm

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>$LLL[x_0]$</td>
</tr>
<tr>
<td>2.</td>
<td>$LRL[x_0]$</td>
</tr>
<tr>
<td>3.</td>
<td>$LRL[x_0]$</td>
</tr>
<tr>
<td>4.</td>
<td>$LLR[x_0]$</td>
</tr>
<tr>
<td>5.</td>
<td>$LRR[x_0]$</td>
</tr>
</tbody>
</table>

Table 5-2: List $\mathcal{S}$ after branch-and-prune algorithm

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>6.</td>
<td>$RRL[x_0]$</td>
</tr>
<tr>
<td>7.</td>
<td>$LRL[x_0]$</td>
</tr>
<tr>
<td>8.</td>
<td>$RRL[x_0]$</td>
</tr>
<tr>
<td>9.</td>
<td>$RLR[x_0]$</td>
</tr>
<tr>
<td>10.</td>
<td>$RRR[x_0]$</td>
</tr>
<tr>
<td>11.</td>
<td>$LRR[x_0]$</td>
</tr>
<tr>
<td>12.</td>
<td>$RLR[x_0]$</td>
</tr>
<tr>
<td>13.</td>
<td>$LRR[x_0]$</td>
</tr>
<tr>
<td>14.</td>
<td>$RRR[x_0]$</td>
</tr>
</tbody>
</table>

Figure 5-3: Binary tree representing lists $\mathcal{S}$ (solid line) and $\mathcal{E}$ (dashed line)

Table 5-1 and Table 5-2 indicate the boxes stored in the lists $\mathcal{S}$ and $\mathcal{E}$ in the order they are stored. The list $\mathcal{S}$ stores the boxes as they appear on the binary tree (Figure 5-3) with the branches more to the left stored first. In the list $\mathcal{E}$ the boxes are stored by the order they were eliminated from list $\mathcal{S}$, because of this criteria the order on list $\mathcal{E}$ has little topological meaning. From the example we can observe that the subpaving $\mathcal{S}$ obtained with this method is not minimal since it stores sibling branches. In this example the list $\mathcal{S}$ consists of boxes of the same size but that is generally not the case. If we used $\varepsilon = 0.6$ instead of $\varepsilon = 1.1$, the result is the one displayed in Figure 5-4 and the list $\mathcal{S}$ has boxes of different sizes.

Figure 5-4: Result of the example using $\varepsilon = 0.6$

The property that only boxes intersecting any box on list $\mathcal{E}$ are eliminated or bisected is useful to reduce the number of boxes to be evaluated during the algorithm, and thus save computational time. To make best use of this property it is necessary that every box on list $\mathcal{E}$ is not intersecting the set $\mathcal{T}$, or else some boxes are being evaluated and bisected when it is not necessary. This property can be observed when enclosing the half circle defined by $(x_1 + 2)^2 + x_2^2 < 4$ $\land x_1 > -2$. To enclose this area may use the following test function:
\[ test([x]) = \begin{cases} \text{false}, & ([x_1] + 2)^2 + [x_2]^2 \geq 4 \\ \text{true}, & \text{otherwise} \end{cases} \]

As input parameters, we can use an initial box \([x_0] = ([−2,2],[−2,2])\) and \(\varepsilon = 0.26\). Note that this is a valid way to apply the algorithm because \(\mathcal{T} \subseteq [x_0]\) and if \(test([x])\) is false \([x] \cap \mathcal{T} = \emptyset\). The algorithm was tested on Matlab™ with an initial list \(\mathcal{J} = (2,−2,2,−2,2,2)\) (Figure 5-5) and with \(\mathcal{J} = (−2,−2,2,2,2,2)\) (Figure 5-6). In the first case the computational time was 0.6270 s and on the second case was 1.1010 s. On both cases the program made 10 iterations. This difference in time is due to the fact that on the first case fewer boxes are evaluated as can be seen on Figure 5-5 and Figure 5-6.

![Figure 5-5: Subpaving \(\mathcal{S}\) obtained with boxes on \(\mathcal{J}\) not in contact with \(\mathcal{T}\)](image)

![Figure 5-6: Subpaving \(\mathcal{S}\) obtained with some boxes on \(\mathcal{J}\) in contact with \(\mathcal{T}\)](image)

The initialization of the list \(\mathcal{E}\) by the set \(\mathcal{J}\) can also influence the overestimation. We can define \(\emptyset\) as the set defined by the union of all the boxes on \(\mathcal{E}\), excluding the ones with what list \(\mathcal{E}\) was initialized (set \(\mathcal{J}\)). On this algorithm only boxes intersecting with any box on the list \(\mathcal{E}\) can be eliminated and then inserted on \(\mathcal{E}\). The union of a compact set with a box in contact with that set is a compact set. Then, by recursion, at any point in the algorithm, the set \(\emptyset\) can be described by the union of a finite number of compact sets, each one intersecting a box that belongs to \(\mathcal{J}\). If we describe \([x]\) as the union of sets not in contact with each other (\(X \text{ and } Y\) are not in contact if \(X \cap cl(Y) = \emptyset \text{ and } cl(X) \cap Y = \emptyset\)). \(cl(X)\) is the closure of \(X\) and we call \(\mathcal{M}\) one of those sets, since \(\emptyset \subset cl([x_0] \setminus \mathcal{T})\), \(\mathcal{M}\) intersects \(\emptyset\) if and only if there is a compact set intersecting a box on the initialization of list \(\mathcal{E}\) included on \(\mathcal{M}\). If \(\mathcal{M}\) does not intersect any box on \(\mathcal{J}\) then it is impossible to find a compact set intersecting a box on \(\mathcal{J}\) included on \(\mathcal{M}\), and then \(\mathcal{M} \cap \emptyset = \emptyset\). If this is the case \(\mathcal{M} \subset S = cl([x_0] \setminus \emptyset)\), and an overestimation occurs because \(\mathcal{M}\) does not belong to \(\mathcal{T}\).
An example where this overestimation occurs is when we enclose a circle with a hole inside defined by $x_1^2 + x_2^2 < 1 \land x_1^2 + x_2^2 > 0.25$. For this example we may use the test function:

$$test([x]) = \begin{cases} 
false, & [x_1]^2 + [x_2]^2 \geq 1 \lor [x_1]^2 + [x_2]^2 \leq 0.25 \\
true, & \text{otherwise}
\end{cases} \quad (5-3)$$

As input parameters, we can use an initial box $[x]_0 = ([-2,2], [-2,2])$ and $\varepsilon = 0.04$. On this case we can describe $[x]_0 \setminus \mathbb{T}$ as the union of two non intersecting sets $\{x \in \mathbb{R}^2: x_1^2 + x_2^2 \geq 1 \land x \in [x]_0\} \cup \{x \in \mathbb{R}^2: x_1^2 + x_2^2 \leq 0.25\}$. We can consider the hole as $\mathbb{M} = \{x \in \mathbb{R}^2: x_1^2 + x_2^2 \leq 0.25\}$. If we define $J = ((-2, [-2,2]), (2, [-2,2]), ([2,2], -2), ([2,2], 2))$ no box is in contact with $\mathbb{M}$, and $\mathbb{M}$ will be included on the subpaving $\mathbb{S}$. The result with this initialization of list $\varepsilon$ is presented on Figure 5-7. As expected the hole is contained by $\mathbb{S}$. If we define $J = ((-2, [-2,2]), (2, [-2,2]), ([2,2], -2), ([2,2], 2), (0,0))$, the box $(0,0)$ is in contact with $\mathbb{M}$ and it is possible that $\mathbb{M}$ will not be included on the subpaving $\mathbb{S}$. With this initialization the result is the one drawn on Figure 5-8. With this initialization it was possible to find an underestimation of the hole.

![Figure 5-7](image1.png) ![Figure 5-8](image2.png)

**Figure 5-7:** Subpaving $\mathbb{S}$ obtained without any box of $J$ in contact with $\mathbb{M}$  
**Figure 5-8:** Subpaving $\mathbb{S}$ obtained with one box of $J$ in contact with $\mathbb{M}$

### Operational Results

The algorithm implemented in Matlab™ was tested to verify its performance on a computer with an Intel® Core™2 Quad CPU at 2.40 GHz. For this effect the program was made to produce overestimations of $n$-spheres for different values of the minimum precision required ($\varepsilon$) and different number of dimensions. The test function is defined as:

$$test([x]) = \begin{cases} 
false, & \sum_{i=1}^{n} [x_i]^2 \geq 1 \\
true, & \text{otherwise} \end{cases} \quad (5-4)$$

The initial box is an $n$-cube centered at the origin and with sides with a length of 4 units, $[x]_0 = ([2,2], \ldots, [-2,2])$. The list $J$ contains all the sides of the initial box $[x]_0$, which is the set defined by:

$$J = \{[x] \in \mathbb{R}^n: i \neq k \Rightarrow [x]_i = [-2,2] \land ([x]_k = -2 \lor [x]_k = 2) \land i, k \in \{1, \ldots, n\} \}. $$

43
In this test, the observed performance parameters were the time to run, the number of boxes on the final result, and the number of processed boxes (the number of times the function $test([x])$ was called). For each number of dimensions used, the algorithm was tested with decreasing values of $\varepsilon = 1.1/2^k$, by increasing the integer number $k$, so that the minimum box size on the obtained subpaving is $1/2^k$. The values of $k$ considered for a specific number of dimensions started at 0 and increased until it reached 53 or, to economize time, until a test took more than 5 minutes to run, or until it reached the maximum value of $k$ obtained on lower dimensions minus one (because in that case the time to run is already known to be longer than 5 minutes). The reason why $k$ cannot exceed 53 is that MATLAB™ generally uses the IEEE 754 binary standard arithmetic, where for a double precision type the format is a 64-bit word divided into a 1-bit sign indicator $s$, an 11-bit biased exponent $e$ and a 52-bit fraction $f$. The real number represented by this format is the following value:

$$\text{value} = \begin{cases} (-1)^s(2^{e-1023})(1.f), & \text{normalized, } 0 < e < 2047 \\ (-1)^s(2^{e-1022})(0.f), & \text{denormalized, } e = 0, f > 0 \\ \text{exceptional value}, & \text{otherwise} \end{cases}$$

(5-5)

Since INTLAB uses MATLAB's arithmetic, next to the value 1 the previous real number available on this format is $1 - 2^{-53}$ with $s = 0$, $e = 1022$ and $f = 2^{53} - 1$, and therefore it is impossible to represent an interval including 1 with width lower than $2^{-53}$. In this example there are boxes being bisected close to the value 1 but always lower than 1, then if we use an $\varepsilon$ smaller than $2^{-53}$ the main cycle of the algorithm will never meet the stopping criteria that all the boxes in contact with any box on the list $E$ have $w([x]) \leq \varepsilon$, and the algorithm will be stuck on an infinite cycle.

The results obtained with this test are in appendix B and displayed in the next figures.

![Figure 5-9: Time to run (seconds) as a function of the minimum box size and number of dimensions](image-url)
Figure 5-10: Number of boxes on the result as a function of the minimum box size and number of dimensions

Figure 5-11: Processed boxes during the algorithm as a function of the minimum box size and number of dimensions

In the previous figures there are also present the points referring to the time to run and number of boxes on the final subpaving for 6 dimensions and $k = 2$, that would not be present following the method described previously. As expected, for $k = 0$ the number of boxes on the final subpaving is $2^n$, where $n$ is the number of dimensions. For $k = 1$ the number of boxes is $4^n$ until $n = 4$ where the 16 “corners” are eliminated from the subpaving because the diagonal of the 4-cube of width $2^{-1}$ is equal to the radius of the 4-ball we are overestimating ($2^{-1} \cdot \sqrt{4} = 1$). As can be easily seen, for 1 dimension the number of boxes on the final result is given by $2(k + 1)$ and the number of processed boxes is given by $2(k + 4)$. Also for 1 dimension, the time to run can be described with very high accuracy as a linear function of the square of the number of processed boxes ($0.0002167pb^2$ where $pb$ is the number of processed boxes), as can be seen in the following graphic.
For higher dimensions such simple and accurate models were not found. For each number of dimensions higher than one it is possible to obtain models using the minimum box size (mbs) as the independent variable for the time to run (tr), number of boxes on the final result (nb) and number of processed boxes (pb). To avoid that the effects which are more related to the instantaneous activity of the microprocessor than with the efficiency of the algorithm are taken into account, the samples where the time to run is lower than 0.5 seconds are not used on the next models. For every number of dimensions the model found to fit the best to on the form \[ a + b \cdot \log(mbs) \]. The coefficients obtained for each model are presented on the following table.

<table>
<thead>
<tr>
<th>Num. of dimensions</th>
<th>Num. of samples</th>
<th>log (tr)</th>
<th>log (nb)</th>
<th>log (pb)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>b</td>
<td>a</td>
<td>( R^2 )</td>
<td>b</td>
</tr>
<tr>
<td>1</td>
<td>-0.0758</td>
<td>-1.6389</td>
<td>0.9873</td>
<td>-0.0395</td>
</tr>
<tr>
<td>2</td>
<td>-2.0257</td>
<td>-3.6289</td>
<td>0.9937</td>
<td>-1.0748</td>
</tr>
<tr>
<td>3</td>
<td>-3.6928</td>
<td>-2.7429</td>
<td>0.9985</td>
<td>-2.2186</td>
</tr>
<tr>
<td>4</td>
<td>-4.6006</td>
<td>-0.5995</td>
<td>0.9757</td>
<td>-3.4160</td>
</tr>
<tr>
<td>5</td>
<td>-5.3138</td>
<td>0.6947</td>
<td>0.9773</td>
<td>-4.3841</td>
</tr>
<tr>
<td>6</td>
<td>-6.7309</td>
<td>1.6555</td>
<td>0.9803</td>
<td>-5.0875</td>
</tr>
<tr>
<td>7</td>
<td>-6.3747</td>
<td>3.0303</td>
<td>1.0000</td>
<td>-6.0000</td>
</tr>
</tbody>
</table>

To obtain a global model for higher dimensions we can fit a linear model to the coefficients on the previous table using the number of dimensions as the independent variable. A good approximation can be made with a linear model for the coefficients \(a\) and a linear function intersecting 0 for \(b\). The obtained models are displayed on the next graphic.
The global models obtained with this method are the following (\(n\) refers to the number of dimensions):

- \( tr \approx e^{1.3653n - 6.409} \cdot mbs^{-1.044n} \)
- \( nb \approx e^{0.5037n + 1.1676} \cdot mbs^{-0.8411n} \)
- \( pb \approx e^{0.9468n + 1.1237} \cdot mbs^{-0.6251n} \)

Beside this method, using the same data plus the samples from dimensions higher than 7, the same models were obtained with the statistics software SPSS™ making a global linear regression with an interaction term:

- \( tr \approx e^{1.2827n - 6.1196} \cdot mbs^{-1.0913n}, \quad R^2 = 0.9712 \)
- \( nb \approx e^{0.7468n - 0.1067} \cdot mbs^{-0.7750n}, \quad R^2 = 0.9454 \)
- \( pb \approx e^{0.9183n + 1.2072} \cdot mbs^{-0.6564n}, \quad R^2 = 0.9659 \)

The next graphics present the predicted values by the model obtained by the linear regression of the coefficients (green line), the model obtained by the global linear regression (blue line) and the observed values.
With the previous analysis we can conclude that the time to run, the number of boxes in the result and the number of processed boxes in the final result increase exponentially with the number of dimensions (the well known “curse of dimensionality”), and decrease less than exponentially with the minimum box size.

From the comparison between the results of the specific models and the global models for the time to run, number of processed boxes and number of boxes on the final subpaving, we can observe that the global model provides a better fit for lower dimensions and the specific model behaves better for higher dimensions. This is due to the fact that the global model is a product of an optimization process that attributes equal weights for each sample, whereas the specific model attributes equal weights for each dimension. Since there are more samples in lower dimensions the global model fits better on lower dimensions.

5.2 Branch and prune (B&P) algorithm with flag

In the algorithm described previously a box with width smaller than $\varepsilon$ continues to be evaluated, even after it was not eliminated from $\mathcal{S}$ on one first evaluation. Because of this fact, the algorithm is suitable to obtain subpavings when the test function does not remain constant throughout the algorithm, for instance when the result from $\text{test}([x])$ depends on the list $\mathcal{S}$, that is, instead of using $\text{test}([x])$ we use $\text{test}([x], \mathcal{S})$. If $\mathcal{S}_i$ is the list $\mathcal{S}$ at some iteration of the algorithm and $\mathcal{S}_{i+1}$ the list $\mathcal{S}$ on the next iteration, from the stopping criteria of the algorithm $\mathcal{S}_{i+1}$ is always different than $\mathcal{S}_i$, then for some boxes $[x]$ with $w([x]) \leq \varepsilon$, one first evaluation of $[x]$ may yield $\text{test}([x], \mathcal{S}_i) = \text{true}$ and on the next iteration $\text{test}([x], \mathcal{S}_{i+1}) = \text{false}$. On that case the box $[x]$ is eliminated.
Although this feature is useful when the test function is of the form \( test([x], S) \), when using test functions that yield the same result for the same box at every iteration we can stop evaluating boxes with \( w([x]) \leq \varepsilon \) after they have been evaluated once, obtaining the same result for a lower number of times the function \( test([x]) \) is run.

In order to deny that boxes with \( w([x]) \leq \varepsilon \) are evaluated twice, a Boolean variable flag is associated with each box and on the next iteration if flag is true the box is not evaluated. The algorithm described previously has thus the following alterations. The first box of \( S \), \([x]_0\), is initialized with an associated flag set to false. Adding to the condition for \([x]\) to be evaluated being when \([x]\) intersects any box on \( E \), also the flag associated with \([x]\) has to be false. The outcomes from the various results of \( test([x]) \) and \( w([x]) \) are altered too. When \( test([x]) \) is false nothing changes from the previous algorithm.

When \( test([x]) \) is true and \( w([x]) > \varepsilon \) the bisected boxes must be evaluated on the next iteration, then the flags associated with \( L[x] \) and \( R[x] \) that are inserted on \( S \) are set to false. When \( test([x]) \) is true and \( w([x]) \leq \varepsilon \) the box \([x]\) is inserted back on \( S \) with an associated flag set to true and thus that box will no longer be evaluated. The fluxogram of the modified algorithm is presented on Figure 5-17.

The previous Branch and Prune algorithm is not suitable for computing bounds for reachable sets since the test function for computing reachable sets, that will be described later, yields the same result for a given box at any step of the algorithm. On the other hand, one efficient way of computing an overestimation of the time independent capture basin, as defined on definition III, involves a test function of the type \( test([x], S) \).

If we are computing the over-approximation of the time independent capture basin, \( C^+ \), using a branch and prune algorithm with a test function based on theorem V, at each iteration the set \( S \), defined by the list of boxes where no conclusion could be made by (4-5) previously on the algorithm, contains the capture basin, that is, \( C \subset S \). Then for the test function based on (4-5) we can use \( S \) as the over-approximation of \( C, C^+ \).

The test function described on the last paragraph was applied as follows:

**Algorithm** test(in: \([x], U, t_{end}, Nstp_{integ}, C^+\); out: Leave)

1. \( Leave := 0 \); \( t_{step} := t_{end}/Nstp_{integ} \); \( t := t_{step} \);
2. \( [x] := [\varphi](t, [x], U) \);
3. if \([x] \cap C^+ \{ Leave := 1 ; return; \}; \)
4. if \( t = t_{end} \) return;
5. \( t := t + t_{step} \); goto 2 ;

The inclusion function \([\varphi](t, [x], U)\) is computed as described on the first section of chapter 4.

Using the test function described above the time independent capture basin was computed for the Zermelo’s problem described earlier. For the target set considered, instead of a circle, was a square \(([−1,1],[−1,1])\). The parameters used for the subpaving were \([x_0] = ([0,5],[−4,4])\), \( \varepsilon = 0.05 \) and
\[ J = \{(5, -4, 4)\} \]. As for the inclusion function of the flow described earlier the parameters used were \( \alpha = 0.001 \) and \( \beta = 0.001 \). For the test function the values used were \( NstP_{\text{integ}} = 256 \) and \( t_{\text{end}} = 20 \).

The set \( \mathbb{C}^+ \) used is the set defined by the subpaving \( S \) at the moment the test function is called.

The Branch and Prune algorithm from last section and from this section were compared solving the problem described above, in order to observe the effects when a function of the form \( \text{test}([x], S) \) is used as a test function. Using the Branch and Prune algorithm from the previous section, the obtained result is presented on Figure 5-18.

![Fluxogram of the Branch and Prune algorithm with flag](image-url)

**Figure 5-17: Fluxogram of the Branch and Prune algorithm with flag**

![Overestimation of time independent capture basin using B&P algorithm](image-url)

**Figure 5-18: Overestimation of time independent capture basin using B&P algorithm**

Using the Branch and Prune algorithm with flag from this section, the obtained result was the following.
From the figures above we can observe that using the Branch and Prune algorithm with flag we obtain a higher overestimation. This is because some boxes on Figure 5-19 with \( w([x]) \leq \varepsilon \) were not eliminated at some point on the B&P algorithm with flag, and if those boxes were tested again further during the execution of algorithm they would have been eliminated, thus allowing other boxes to be eliminated and so forth.

For the objective of this thesis, which is the computation of bounds in reachable sets, only test functions of the form \( test([x]) \) are used. As was already discussed, on such cases it is more efficient to use the Branch and Prune algorithm with flag. For this reason, only the Branch and Prune algorithm with flag will be considered for the rest of this work.

At this point it is important to note that, from the point of view of the resulting set, considering the subpaving obtained with the SIVIA algorithm described in the last chapter, and the subpaving obtained with the Branch and Prune with flag algorithm, if we describe \([x]_0 \setminus S_{SIVIA}\) as the union of sets not in contact with each other and if all those sets are in contact with a box on set \( J \), then \( S_{SIVIA} = S_{B&Pwf} \), otherwise we have \( S_{SIVIA} \subset S_{B&Pwf} \) and we have another source of overestimation. That is, there can be an overestimation caused by the initialization of \( J \) not being conservative (This fact was already seen for the Branch and Prune algorithm of the last section. As was also discussed on the last section, if \( J \) is selected to be too conservative it can have a detrimental effect on the performance). On the cases where \([x]_0 \setminus S_{SIVIA}\) is a compact set and intersects a box on \( J \) there is no overestimation caused by the selection of \( J \) and the result is the same for SIVIA and for the Branch and Prune algorithm.

**Operational Results**

As was done for the first algorithm the performance of the modified algorithm was tested using the test function (4-10), initial boxes defined by \([x]_0 = ([\pm 2],\cdots,[\pm 2])\), and a list \( J \) containing the sides of the initial box, which is the following set:

\[
J = \{ [x] \in \mathbb{R}^n : i \neq k \Rightarrow [x_i] = [-2,2] \land ([x_k] = -2 \lor [x_k] = 2), i, k \in \{1, \cdots, n\} \}
\]

The sampling sequence is the same used for testing the first algorithm. For the number of dimensions tested the values of \( \varepsilon \) used were defined by \( \varepsilon = 1.1/2^k \), where \( k \) is an integer number. The minimum
for each dimension is 0 and the maximum \( k \) is the minimum between the maximum \( k \) used for the last number of dimensions, if the number of dimensions is higher than one, the value of \( k \) such that the algorithm took more than 5 minutes to run, and 52.

The obtained results of number of processed boxes (number of times \( test([x]) \) is called) and time to run of the algorithm from the performance test are presented on Figure 5-23 and on a table of appendix B. The number of boxes on \( S \) is not presented because it is the same as for the Branch and Prune algorithm. The results, for dimensions higher than one, of time to run and processed boxes from the previous algorithm (Branch and Prune algorithm) and the modified algorithm (Branch and Prune algorithm with flag) are displayed on Figure 5-22 to convey a better notion of the differences. It is observable that the time to run is slightly less and the number of processed boxes is slightly lower for the Branch and Prune algorithm with flag than for the previous Branch and Prune algorithm.

As on the previous algorithm, for 1 dimension there is a linear relation between \( k \) and the number of processed boxes, on this case is \( 7 + 2k \), one less than with the previous algorithm. Also for 1 dimension a linear relation makes a good fit between the square of the number of processed boxes and the time to run as can be seen on the graphic of Figure 5-20. The time to run in seconds is approximately \( 0.0002191 \times pb^2 \), where \( pb \) is the number of processed boxes.

![Figure 5-20: Relation between processed boxes and time to run for 1 dimension](image)

![Figure 5-21: Comparison of time to run between B&P (blue) and B&P with flag (green)](image)
Figure 5-22: Comparison of processed boxes between B&P (blue) and B&P with flag (green)

Figure 5-23: Time to run for the modified algorithm (left) and number of processed boxes (right)

Again, for dimensions higher than one there are not such simple relations between the minimum box size, the time to run and number of processed boxes. Models of the form the form $a + b \cdot \log(mbs)$ were obtained for every number of dimensions. The coefficients obtained for every dimension are presented on Table 5-4.

<table>
<thead>
<tr>
<th>Num. of dimensions</th>
<th>Num. of samples</th>
<th>$\log(\text{tr})$</th>
<th>$R^2$</th>
<th>$\log(\text{pb})$</th>
<th>$R^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$b$</td>
<td>$a$</td>
<td></td>
<td>$b$</td>
<td>$a$</td>
</tr>
<tr>
<td>1</td>
<td>-0.0774</td>
<td>-1.6907</td>
<td>0.9905</td>
<td>3.4102</td>
<td>-0.0373</td>
</tr>
<tr>
<td>2</td>
<td>-2.0951</td>
<td>-4.0831</td>
<td>0.9980</td>
<td>3.1346</td>
<td>-1.0738</td>
</tr>
<tr>
<td>3</td>
<td>-3.6685</td>
<td>-2.8540</td>
<td>0.9962</td>
<td>3.6555</td>
<td>-2.1496</td>
</tr>
<tr>
<td>4</td>
<td>-4.6150</td>
<td>-0.8367</td>
<td>0.9719</td>
<td>4.8423</td>
<td>-2.7835</td>
</tr>
<tr>
<td>5</td>
<td>-5.2454</td>
<td>0.5237</td>
<td>0.9715</td>
<td>5.7856</td>
<td>-3.2912</td>
</tr>
<tr>
<td>6</td>
<td>-4.8181</td>
<td>1.9242</td>
<td>1.0000</td>
<td>6.7968</td>
<td>-3.1714</td>
</tr>
<tr>
<td>7</td>
<td>-6.2022</td>
<td>2.9003</td>
<td>1.0000</td>
<td>7.6241</td>
<td>-3.7286</td>
</tr>
</tbody>
</table>

Table 5-4: Coefficients of models for Branch and Prune algorithm with flag

As was done with the Branch and Prune algorithm, to obtain a global model we fit a linear model to the coefficients on Table 5-4 using the number of dimensions as the independent variable. The obtained models are displayed on Figure 5-24.
The global models obtained with this method are the following:

- \[ tr \approx e^{1.4461n - 6.9115 \text{mbs} - 0.9511n} \]
- \[ pb \approx e^{0.9376n + 1.0875 \text{mbs} - 0.585n} \]

As was done previously, the same data used to obtain the specific model plus the samples from dimensions higher than 7 were used to obtain a global model of the behavior of the algorithm with the statistics software SPSS™, by performing a global linear regression with an interaction term:

- \[ tr \approx e^{1.2327n - 5.9335 \text{mbs} - 1.0486n}, \quad R^2 = 0.9452 \]
- \[ pb \approx e^{0.9165n + 1.1454 \text{mbs} - 0.6204n}, \quad R^2 = 0.9554 \]

From the coefficients of both models it is possible to conclude that the “curse of dimensionality” is slightly less severe for the Branch and Prune algorithm with flag than for the previous Branch and Prune algorithm. On the specific model, the coefficient for the exponential growth with dimension and minimum box size (the dominant term) of the time to run of the B&P algorithm with flag is \(-0.9511\), and the coefficient for the exponential growth of the number of processed boxes of the B&P algorithm with flag is \(-0.585\). Those coefficients are higher than, respectively, the coefficient for the exponential growth of the time to run of the previous B&P algorithm \((-1.044)\) and the coefficient for the exponential growth of the number of processed boxes with time \((-0.6251)\). Since the precision of the algorithm is higher for lower values of \text{mbs}, higher values of those coefficients represents a mitigation of the “curse of dimensionality”. The same effect is also observable on the global model. For the B&P algorithm with flag the coefficient for the exponential growth of the time to run is \(-1.0488\) and the coefficient for the exponential growth of the number of processed boxes is \(-0.6204\). Those coefficients are higher than the equivalent coefficients for the previous B&P algorithm, which are respectively \(-1.0913\) and \(-0.6564\).

### 5.3 Improving Computational Efficiency

Using the MATLAB™ code profiler, several possible programming improvements of the computational efficiency of the algorithms described earlier were identified. Since the identified improvements imply fundamental changes in the programming, before devising other algorithms those improvements were implemented. The identified improvements are summarized in the following list.
- **Use matrix notation whenever possible:**
  Using cycles in MATLAB™ unlike in C++ is computationally expensive. A much more efficient alternative in MATLAB™ is matrix notation. For instance, instead of using the expression "for \( i = 1:N \), \( E(i) = 2^i \); end" we should use "\( E(1:N) = 2.^(1:N) \)."

- **Only make alterations on a vector or matrix when absolutely necessary:**
  Altering the size of a vector or a matrix is a computationally expensive process. Therefore a program should resort to modify a vector or a matrix as few as possible.

For instance, when programming the previous algorithms each time a box was tested its place on the list of boxes \( S \), a cell array, was freed and, if the box was not eliminated, the box would be inserted on the first free entry of the list, and if the box was bisected the two children boxes would be inserted on the first two free entries of the list.

A more efficient process would be to only free the place of a box in the list \( S \) only if the box is eliminated. If a box is bisected the left child would remain in the same place as the parent box and the right child would go to the first free index.

- **Store indices of neighbor boxes for each box:**
  To improve the efficiency of the process of determining if a box is in contact with any box on list \( E \), for each box on \( S \) we can store the indices of the boxes on \( S \) it is in contact with, we call that list of indices \( \text{NeighS} \), and store a list of indices of boxes on \( E \) that it potentially intersects, we call that list of indices \( \text{NeighE} \). Determining if a box \([x]\) is in contact with any box on \( E \) amounts to determine whether \([x]\) intersects with any box pointed by its associated list \( \text{NeighE} \). Also, on that process, \( \text{NeighE} \) is refined.

If a box \([x]\) is eliminated the index where \([x]\) was stored on \( E \) is inserted on the list \( \text{NeighE} \) of each box indicated by \( \text{NeighS} \) of \([x]\), and the index where \([x]\) was stored on \( S \) is removed of the list \( \text{NeighS} \) of each box indicated by the list \( \text{NeighS} \) of \([x]\).

If a box is bisected the children inherit the list \( \text{NeighE} \) associated with the parent, and each children has on its associated list \( \text{NeighS} \) the indexes of the boxes pointed by the list \( \text{NeighS} \) of the parent that are determined to intersect with that children.

While this process has the disadvantage of having to manipulate more variables, when dealing with a large number of boxes, reducing the number of candidate boxes on \( E \) a box on \( S \) may intersect contributes largely to the speed of the algorithm.
• Use structures of vectors instead of vectors of structures:
  
  Since on a structure on MATLAB™ the contents stored in memory are the elements of the structure and its header, to reduce the number of headers to manipulate it is more efficient to use a structure as “S.contents.neigh_E[i]” instead of “S.contents[i].neigh_E”.

The improvements described above were implemented on the Branch and Contract algorithm that will be described later. To benefit from the improvements described on this chapter the Branch and Contract algorithm can be made to work as the Branch and Prune algorithm with flag using the following contractor.

\[ C_{\Delta}(x) = \begin{cases} [x], & \text{if } test([x]) = true \\ \emptyset, & \text{if } test([x]) = false \end{cases} \]  

(5-6)

Operational Results

To observe the effect of the improvements described in this section when compared to the B&P algorithm implemented before on MATLAB™, the performance of the Branch and Contract algorithm made to work as a Branch and Prune algorithm was tested using the same sampling sequence of minimum box sizes and number of dimensions as with the Branch and Prune algorithm with flag. The time to run for each sample is displayed on Figure 5-25. Since there are some extra data points, although the result is the same as for the B&P algorithm with flag, the number of processed boxes for each sample are displayed on Appendix B.

![Figure 5-25: Time to run for the improved algorithm](image)

The comparison of the times to run (for dimensions higher than one) between the B&P algorithm implemented before on MATLAB™ (Before alterations) and the Branch and Contract algorithm made to work as a Branch and Prune algorithm (After alterations) is displayed on Figure 5-26.
It is observable from the figure above that for a high minimum box size the time to run after the improvements is higher but for lower values of minimum box size (higher precision) the time to run is lower after the improvements which is the desired outcome.

For one dimension for the improved algorithm instead of a quadratic relation the time to run and the number of boxes as found on the previous algorithms, a linear relation $tr = 0.0118pb - 0.0793$ was found to fit the data better, as is shown on Figure 5-27.

For dimensions greater than one and for times to run greater than 0.5 it is also useful to establish a relation between the time to run and the number of processed boxes, since, observing the algorithm, the number of processed boxes seems to be a variable that directly affects the time to run. The relation obtained is $tr = e^{-6.8776}pb^{1.3425}$ as is shown on Figure 5-28.
5.4 Branch and Contract (B&C)

In this section an algorithm, based on the B&P algorithm with flag described before, of computing subpavings using contractors will be discussed, that algorithm will be named Branch and Contract (B&C).

First we need to define the function \( \text{exclude}([x], [y]) \) that computes a subpaving of \( \text{cl}([y] \setminus [x]) \). The function is defined as follows:

\[
\text{exclude}([x], [y]) = \begin{cases} 
[y], & \text{if } [x] \cap [y] = \emptyset \\
\emptyset, & \text{if } [y] \subset [x]
\end{cases}
\]

\[
\subset \{ [z] \in \mathbb{R}^n | \forall i \in \{1, ..., n\}, \left( [z_i] = [y_i, \max(y_{i, x_i})] \right) \lor \left( [z_i] = \left[ \max(y_{i, x_i}, \min(y_{i, \bar{x}})) \right] \lor \left( [z_i] = \left[ \min(y_{i, \bar{x}}, \bar{x_i}) \right], [z] \neq [x] \cap [y], w([z]) > 0 \right) \}, \text{ otherwise}
\]

In the Branch and Contract algorithm, we use a contractor \( C^*_{\text{B}}(\{x\}) \subset [x] \cap T \), which is the term \( [x^N] \) of a sequence \( [x^{k+1}] = C_T([x^k]) \), under some stopping criteria. The stopping criteria considered were defining some \( sc \) and the series stops at \( N = sc \) or the series stops when \( w([x^{N-1}]) - w([x^N]) < sc \), on the last case \( sc \) is a measure of the required precision. As inputs, the algorithm must take an initial over approximation of the set \( \text{cl}([x_0]) \) (which will be the root of the subpaving \( S \)), the initial outer set of boxes in \( E \) and the parameters used in the test.

The B&C algorithm is similar to the B&P algorithm with flag, but has some significant alterations and therefore it will be described integrally here. The B&C algorithm is described in the flow chart of Figure 5-29. First the list \( S \) is initialized with \( [x_0] \) and the list \( E \) is initialized with boxes in contact with \( [x_0] \) and preferably not in contact with the set \( T \). We will call the set containing those initial boxes \( J \). List \( E \) will also contain the boxes that form a subpaving of \( \text{cl}([x_0] \setminus S) \) at a given point on the algorithm.

The initial number of boxes on \( S \) (NB) is one. The algorithm keeps refining the subpaving until every box on \( S \) intersecting a box on \( E \) has \( \text{flag}([x]) = true \). This is implemented by initializing ‘Check again’ as false and if any alteration on \( S \) occurs during the iteration ‘Check again’ becomes true. On each iteration every box on \( S \) must be checked. This consists of a cycle that repeats itself for ‘initial NB’ times, which is the number of boxes on \( S \) (NB) at the beginning of the iteration. The cycle starts by removing the first box from \( S \) and attributing it to \( [x] \). If the box \( [x] \) does not intersect any box on \( E \), or \( \text{flag}([x]) = true \), then \( [x] \) should stay on \( S \) and the box is inserted on the last entry of \( S \).

If the box \( [x] \) intersects any box on \( E \), or \( \text{flag}([x]) = false \) it should be contracted under \( C^*_{\text{T}}([x]) \) and the result will be denoted as \( [x]_{\text{new}} \). If the result is empty, \( [x]_{\text{new}} = \emptyset \) the box does not intersect the set \( T \) and therefore it should not belong in the subpaving \( S \). If this is the case, the box \( [x] \) is inserted as the last entry of list \( E \). On this case list \( S \) loses an element so NB should be subtracted by one and ‘Check again’ should become true. If \( [x]_{\text{new}} = \emptyset \), \( [x]_{\text{new}} \) should remain on the subpaving and the list resulting from \( \text{exclude}([x], [x]_{\text{new}}) \) should be inserted on \( E \).
If this is the case there are two possible outcomes. Or \( w([x]_{\text{new}}) > \varepsilon \) and \([x]_{\text{new}}\) should be bisected, or \( w([x]_{\text{new}}) \leq \varepsilon \) and it is not necessary to evaluate \([x]_{\text{new}}\) any more. On the first case \([x]_{\text{new}}\) is bisected and \(L[x]_{\text{new}}\) and \(R[x]_{\text{new}}\) are inserted on \(S\) instead of \([x]_{\text{new}}\). First \(L[x]_{\text{new}}\) is inserted, with \(\text{flag}(L[x]_{\text{new}}) = \text{false}\), as the last entry of \(S\) and then \(R[x]_{\text{new}}\) is inserted, with \(\text{flag}(L[x]_{\text{new}}) = \text{false}\), as the last entry of \(S\) after \(L[x]_{\text{new}}\). On this case, a box is bisected on \(S\), then NB should be increased by one and ‘Check again’ becomes true. On the second case, box \([x]_{\text{new}}\) should remain on \(S\) then it is stored as the last entry on \(S\), with \(\text{flag}([x]_{\text{new}}) = \text{true}\). Since on this case list \(S\) does not change, NB remains the same and ‘Check again’ remains the same.

Figure 5-29: Flowchart of the branch-and-contract algorithm

As an example of the formation of the subpaving on the B&C algorithm, as was done for the B&P algorithm, to enclose the circle defined as \((x_1 - 0.5)^2 + (x_2 - 0.5)^2 < 2.25\) we may use the following contractor.

\begin{aligned}
\text{Algorithm } C_T \ (\text{inout: } [x]) \\
\text{1} & \quad \text{[sq]} := ([x] - (0.5, 0.5))^2 \\
\text{2} & \quad \text{R} := [0, 2.25] \cap \sum_{i=1}^{N}[sq_i] \\
\text{3} & \quad j := 1 \\
\text{4} & \quad \text{Do} \\
\text{5} & \quad \quad [sq] := \text{R} \cap \sum_{i=1}^{N}[sq_i] \\
\text{6} & \quad \quad \text{While } j \leq N \\
\text{7} & \quad \quad \quad [x] := (([x] \cap \sqrt{[sq]}) \cup ([x] \cap -\sqrt{[sq]}]) + (0.5, 0.5)
\end{aligned}
As input parameters, we can use the initial box \([x]_0 = ([\text{v}2,2], [-2,2])\), an initial list \(I = ([\text{v}2,2], 2, [-2,2], [-2,2], [-2,2], 2, [-2,2], 2)\) and \(\epsilon = 0.5\). The stopping criteria for \(C^*T([x])\) is \(w([x^{N-1}]) - w([x^0]) \leq 10^{-10}\). The result is drawn on Figure 5-30 and Figure 5-31. The circle in blue represents the boundary of the set \(T\).

![Figure 5-30: Evolution of list \(\delta\)](image)

![Figure 5-31: Final outer subpaving \(\mathcal{O}\)](image)

![Figure 5-32: Binary tree representing list \(\delta\)](image)

Table 5-5 and Table 5-6 indicate the boxes stored on the lists \(S\) and \(E\) in the order they are stored. The list \(S\) stores the boxes as they appear on the binary tree (Figure 5-32) with the branches more left stored first. On the list \(E\) the boxes are stored by the order they were eliminated from list \(S\). From the example we can observe that the subpaving \(S\) obtained with this method is not regular since each box is a product of successive bisections and contractions. On the binary tree we are considering the children of a box \([x]\) as \(LC_T([x])\) and \(RC_T([x])\) and therefore the union of both children may not yield the parent box, that is, \(LC_T([x]) \cup RC_T([x]) = C_T([x]) \subseteq [x]\).
Table 5-5: List $\mathcal{E}$ after branch-and-contract algorithm

<table>
<thead>
<tr>
<th>$\mathcal{E}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(-2,[-2,2])$</td>
</tr>
<tr>
<td>$(2,[-2,2])$</td>
</tr>
<tr>
<td>$([-2,2],-2)$</td>
</tr>
<tr>
<td>$([-2,2],2)$</td>
</tr>
</tbody>
</table>

17. $\text{exclude}(\mathcal{C}_T^r(\{x_0\}), \{x_0\})$

18. $\text{exclude}(\mathcal{C}_T^r(\{LLLC^r_\tau(\{x_0\})\}), \{LLLC^r_\tau(\{x_0\})\})$

19. $\text{exclude}(\mathcal{C}_T^r(\{LRLC^r_\tau(\{x_0\})\}), \{LRLC^r_\tau(\{x_0\})\})$

20. $\text{exclude}(\mathcal{C}_T^r(\{RLRC^r_\tau(\{x_0\})\}), \{RLRC^r_\tau(\{x_0\})\})$

21. $\text{exclude}(\mathcal{C}_T^r(\{RRRC^r_\tau(\{x_0\})\}), \{RRRC^r_\tau(\{x_0\})\})$

22. $\text{exclude}(\mathcal{C}_T^r(\{LC^r_\tau(\{LLLC^r_\tau(\{x_0\})\})\}), \{LC^r_\tau(\{LLLC^r_\tau(\{x_0\})\})\})$

23. $\text{exclude}(\mathcal{C}_T^r(\{RC^r_\tau(\{LRLC^r_\tau(\{x_0\})\})\}), \{RC^r_\tau(\{LRLC^r_\tau(\{x_0\})\})\})$

24. $\text{exclude}(\mathcal{C}_T^r(\{LC^r_\tau(\{RLRC^r_\tau(\{x_0\})\})\}), \{LC^r_\tau(\{RLRC^r_\tau(\{x_0\})\})\})$

25. $\text{exclude}(\mathcal{C}_T^r(\{RC^r_\tau(\{RRRC^r_\tau(\{x_0\})\})\}), \{RC^r_\tau(\{RRRC^r_\tau(\{x_0\})\})\})$

<table>
<thead>
<tr>
<th>$\mathcal{S}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1. \mathcal{C}<em>T^r({LLLC^r</em>\tau({x_0})})$</td>
</tr>
<tr>
<td>$2. \mathcal{C}<em>T^r({LRLC^r</em>\tau({x_0})})$</td>
</tr>
<tr>
<td>$3. \mathcal{C}<em>T^r({RLRC^r</em>\tau({x_0})})$</td>
</tr>
<tr>
<td>$4. \mathcal{C}<em>T^r({RRRC^r</em>\tau({x_0})})$</td>
</tr>
<tr>
<td>$5. \mathcal{C}<em>T^r({LC^r</em>\tau({LLLC^r_\tau({x_0})})})$</td>
</tr>
<tr>
<td>$6. \mathcal{C}<em>T^r({RC^r</em>\tau({LRLC^r_\tau({x_0})})})$</td>
</tr>
<tr>
<td>$7. \mathcal{C}<em>T^r({RC^r</em>\tau({RLRC^r_\tau({x_0})})})$</td>
</tr>
<tr>
<td>$8. \mathcal{C}<em>T^r({RRRC^r</em>\tau({x_0})})$</td>
</tr>
<tr>
<td>$9. \mathcal{C}<em>T^r({LLLC^r</em>\tau({x_0})})$</td>
</tr>
<tr>
<td>$10. \mathcal{C}<em>T^r({LRLC^r</em>\tau({x_0})})$</td>
</tr>
<tr>
<td>$11. \mathcal{C}<em>T^r({RLRC^r</em>\tau({x_0})})$</td>
</tr>
<tr>
<td>$12. \mathcal{C}<em>T^r({RRRC^r</em>\tau({x_0})})$</td>
</tr>
<tr>
<td>$13. \mathcal{C}<em>T^r({LC^r</em>\tau({LLLC^r_\tau({x_0})})})$</td>
</tr>
<tr>
<td>$14. \mathcal{C}<em>T^r({RC^r</em>\tau({LRLC^r_\tau({x_0})})})$</td>
</tr>
<tr>
<td>$15. \mathcal{C}<em>T^r({RC^r</em>\tau({RLRC^r_\tau({x_0})})})$</td>
</tr>
<tr>
<td>$16. \mathcal{C}<em>T^r({RRRC^r</em>\tau({x_0})})$</td>
</tr>
</tbody>
</table>

Table 5-6: List $\mathcal{S}$ after branch-and-contact algorithm

Since when $[x] \setminus \mathcal{C}_T^r(\{x\}) \neq \emptyset$ it may happen that $[x] \cap [y] \neq \emptyset$ but $[y] \cap [x] \setminus \mathcal{C}_T^r(\{x\}) = \emptyset$, the fundaments leading to the assumption that on the B&P algorithm if we describe $[x]\setminus T$ as the union of sets not in contact with each other, and we call $\mathbb{M}$ one of those sets, then $\mathbb{M}$ intersects $\mathcal{O}$ only if $\mathbb{M}$ intersects a box on $\mathcal{J}$, are no longer valid. Depending on the process of formation of $\mathcal{O}$ it may happen that $\mathbb{M} \cap \mathcal{O} \neq \emptyset$. It must be advised that, that property is not to be relied upon since it depends on the process of formation of $\mathcal{O}$, there may be cases where $\mathbb{M}$ is a significant area of $[x_0]\setminus T$ and it remains on $\mathcal{S}$, $\mathbb{M} \subset \mathcal{S} = cl(\{x_0\} \setminus \mathcal{O})$, if there is not a box on $\mathcal{J}$ that intersects $\mathbb{M}$.

An example where this overestimation may not occur is when we enclose a circle with a hole inside defined by $x_1^2 + x_2^2 < 1 \land x_1^2 + x_2^2 > r_m^2$. For this example we may use the contractor:

**Algorithm** $\mathcal{C}_T$ (inout: $[x]$)

1. $[sq] := [x]^2$
2. $R := [r_m^2, 1] \cap \sum_{i=1}^{N}[s_i q_i]$
3. $j := 1$
4. Do
5. $[sq] := R \cap \sum_{i\neq j}[s_i q_i]$
6. While $j \leq N$
7. $[x] := [(x) \cap \sqrt{[sq]}] \cup ([x] \cap -\sqrt{[sq]})$
As input parameters, we can use an initial box \([x]_0 = ([-2.2], [-2.2])\) and \(\epsilon = 0.04\). On this case we can describe \([x]_0 \setminus T\) as the union of two non intersecting sets \(\{x \in \mathbb{R}^2 : x_1^2 + x_2^2 \geq 1 \land x \in [x]_0\} \cup \{x \in \mathbb{R}^2 : x_1^2 + x_2^2 \leq r_{in}^2\}\). We can consider the hole as \(M = \{x \in \mathbb{R}^2 : x_1^2 + x_2^2 \leq r_{in}^2\}\). If we define \(J = \{(-2, [-2,2]), (2, [-2,2]), ([-2,2], -2), ([-2,2], 2)\}\) no box is in contact with \(M\), and \(M\) may be included on the subpaving \(S\). The resulting subpaving \(S\) when \(r_{in}^2 = 0.25\) is displayed on Figure 5-33 and the B&C algorithm failed to find the hole \(M\). The result when \(r_{in}^2 = 0.5041\) is displayed on Figure 5-34. On the last case it is observable that the hole \(M\) was detected by the B&C algorithm. This effect is due to the fact that during the algorithm the box \([x] = ([-0.5,0], [-1,0])\) is evaluated. On the first case \(C_T([x]) = [x]\) but on the second case \(\left([x] \setminus C_T([x])\right) \cap M \neq \emptyset\) and since \([x] \setminus C_T([x])\) is inserted on \(\epsilon\) the hole \(M\) is detected.

![Figure 5-33: Subpaving \(S\) obtained with \(r_{in}^2 = 0.25\)](image)

![Figure 5-34: Subpaving \(S\) obtained with \(r_{in}^2 = 0.5041\)](image)

**Operational Results**

The Branch and contract algorithm was implemented on Matlab™ and tested in order to verify its performance, on a computer with an Intel® Core™2 Quad CPU at 2.40 GHz. As was done for the Branch and Prune algorithms, the program was made to produce overestimations of n-spheres for different values of the minimum precision required (\(\epsilon\)) and different number of dimensions. The used contractor is defined as the following algorithm:

```plaintext
Algorithm \(C_T\) (inout: \([x]\))
8 \([sq] := [x]^2\)
9 \(R := [0,1] \cap \Sigma_{i=1}^N [sq_i]\)
10 \(j := 1\)
11 \(\text{Do}\)
12 \([sq] := R \cap \Sigma_{i \neq j} [sq_i]\)
13 \(\text{While } j \leq N\)
14 \([x] := ([x] \cap \sqrt{[sq]}) \cup ([x] \cap -\sqrt{[sq]})]\)
```

Where \([x]^2\) and \(\sqrt{[x]}\) are defined as the operation applied to each component, that is, \(([x]^2)_i = [x_i]^2\) and \((\sqrt{[x]})_i = \sqrt{[x_i]}\). The stopping criteria used was \(w([x^{k-1}]) - w([x^k]) \leq 10^{-10}\).

The set-up for the performance test is very similar to the previous performance tests with a slight difference on \(\epsilon\). The initial box is an n-cube centered at the origin and with sides with a length of 4
units, \([x]_0 = (\ldots, [-2, 2], \ldots, [-2, 2])\). The list \(J\) contains all the sides of the initial box \([x]_0\), which is the set defined by:

\[
J = \{[x] \in \mathbb{R}^n \mid i \neq k \Rightarrow [x_i] = [-2, 2] \wedge ([x_k] = 2) \vee [x_k] = 2), i, k \in \{1, \ldots, n\}\}.
\]

On this test, the observed performance parameters were the time to run, the number of boxes on the final result, and the number of processed boxes (the number of times \(C'_T([x])\) was called). For each number of dimensions used, the algorithm was tested with decreasing values of \(\varepsilon = 1/2^k\), by increasing the integer number \(k\). The values of \(k\) considered for a specific number of dimensions started at 0 and increased until it reached 52 or, to economize time, until a test took more than 5 minutes to run, or until it reached the maximum value of \(k\) obtained on lower dimensions minus one.

The resulting times to run, number of boxes on the final subpaving and number of processed boxes are displayed on Figure 5-37, Figure 5-38 and Figure 5-39 respectively.

For one dimension it was found that a linear relation between the time to run and the number of processed boxes describes very well the performance of the B&C algorithm, \(tr \approx 0.0137pb\), as is shown on Figure 5-35.

![Figure 5-35: time to run for one dimension vs. processed boxes](image)

For a number of dimensions greater than one and a time to run greater than 0.5, the relation \(tr \approx e^{-5.0018}pb^{1.2472}\) seems to describe the performance of the algorithm with a good accuracy as is shown on Figure 5-36.

![Figure 5-36: logarithm of time to run vs. logarithm of processed boxes](image)
Figure 5-37: Time to run (seconds) as a function of the minimum box size and number of dimensions

Figure 5-38: Number of boxes on the result as a function of the minimum box size and number of dimensions

Figure 5-39: Processed boxes during the algorithm as a function of the minimum box size and number of dimension

To better capture the behavior of the algorithm, models of the form \( a + b \cdot \log(mbs) \) of \( \log(nb) \) and \( \log(pb) \) were obtained for every number of dimensions. The coefficients obtained for every dimension are presented on Table 5-7.
As was done with the Branch and Prune algorithm, to obtain a global model we fit a linear model to the coefficients on Table 5-7 using the number of dimensions as the independent variable. The obtained models are displayed on Figure 5-40.

The global models obtained with this method are the following:

- \( nb \approx e^{0.4517n+1.3117}\ \text{mb}_s^{-1.2201n+1.3872} \)
- \( pb \approx e^{0.4989n+1.7506}\ \text{mb}_s^{-1.2213n+1.3964} \)

As was done previously, the same data used to obtain the specific model plus the samples from dimensions higher than 6 was used to obtain a global model of the behavior of the algorithm with the statistics software SPSS™, by performing a global linear regression with an interaction term:

- \( nb \approx e^{0.4963n+1.6576}\ \text{mb}_s^{-0.9709n+0.8891}, \ R^2 = 0.9723 \)
- \( pb \approx e^{0.5519n+1.9178}\ \text{mb}_s^{-0.99n+0.9197}, \ R^2 = 0.9853 \)

\[
\begin{array}{cccccc}
\text{Num. of dimensions} & \text{Num. of samples} & \log (\text{nb}) & \log (\text{pb}) \\ 
1 & 34 & -0.0414 & 3.2326 & 0.9838 & -0.0408 & 3.2624 & 0.9843 \\ 
2 & 10 & -1.0806 & 2.3735 & 0.9949 & -1.0578 & 2.8464 & 0.9966 \\ 
3 & 4 & -2.2186 & 2.6059 & 0.9993 & -2.2609 & 3.2385 & 0.9977 \\ 
4 & 4 & -3.5632 & 2.9022 & 0.9982 & -3.5243 & 3.5715 & 0.9979 \\ 
5 & 3 & -4.6263 & 3.5521 & 0.9978 & -4.6130 & 4.2324 & 0.9977 \\ 
6 & 2 & -5.9773 & 4.1589 & 1.0000 & -5.9884 & 4.8442 & 1.0000 \\ 
\end{array}
\]

Table 5-7: Coefficients of models for Branch and Contract algorithm

\begin{figure}
\centering
\includegraphics[width=\textwidth]{figure5-40.png}
\caption{Coefficients of models for B&C as a function of the number of dimensions}
\end{figure}

### 5.5 Outer bound

To describe the application of the B&P or B&C algorithms to compute outer bounds on reachable sets we must denote the B&P algorithm shown on Figure 5-17 as the function \( S = B&P(\text{test}(), [x]_o, \epsilon, \mathcal{J}) \), and the B&C algorithm as the function \( S = B&C(C_T(), [x]_o, \epsilon, \mathcal{J}) \).
The test function used on the B&P algorithm to compute an outer bound of reachable set is the following.

\[ test([x]) = \begin{cases} 1, & \text{if } [\varphi](r, [x], [\mathbb{U}], d) \cap \mathcal{T} \neq \emptyset \\ 0, & \text{if } [\varphi](r, [x], [\mathbb{U}], d) \cap \mathcal{T} = \emptyset \end{cases} \quad (5-7) \]

Since it is useful to test several disturbance inputs \( d \in \mathbb{D} \subset \mathbb{R}^p \) on the B&P algorithm, we use, instead of \( test([x]) \), \( test_1([x]) \land \ldots \land test_L([x]) \) where \( test_j([x]) \) is the test function (5-7) with a disturbance input \( d^i \) instead of \( d \). The samples \( d^i \) are selected and are selected considering a number of samples for each dimension ‘ns’ yielding a total number of samples \( L = ns^p \). The sampling set is as defined on (5-8).

\[ \mathcal{D} = \left\{ d \in \mathbb{R}^p ; d_i = \frac{B_{r}^{j-1} + B_{r}(ns-j)}{ns-1}, i \in \{1, \ldots, p\}, j \in \{1, \ldots, ns\} \right\} \quad (5-8) \]

To improve efficiency, since \( \mathcal{T} \), is described through a list of boxes, an associated list was created for every box on the subpaving and every sampled disturbance input \( d^i \in \mathcal{D} \), such that when performing \( [\varphi](r, [x], [\mathbb{U}], d^i) \cap \mathcal{T} \) the associated list of \( [x] \) and \( d^i \) stores a pointer for every box on \( \mathcal{T} \) that intersects with \( [\varphi](r, [x], [\mathbb{U}], d^i) \), and when bisecting \( [x] \) its children inherit its associated lists. With that associated list when computing the intersection only the boxes of the target set pointed by the associated list have to be considered.

The contractor used on the B&C algorithm to compute an outer bound of reachable sets is the following:

\[ C_{\mathcal{T}}([x]) = [x] \cap [\varphi_{inv}](r, [[\varphi](r, [x], [\mathbb{U}], d) \cap \mathcal{T}], [\mathbb{U}], d) \quad (5-9) \]

Also on the B&C algorithm, since it is useful to test several disturbance inputs \( d \in \mathbb{D} \subset \mathbb{R}^p \), instead of contracting \( [x] \) under \( C_{\mathcal{T}}([x]) \), we contract \( [x] \) under the combined contractor \( C_{\mathcal{T}_L} \left( C_{\mathcal{T}_{r=1}} \left( \ldots C_{\mathcal{T}_1}([x]) \right) \right) \) where \( C_{\mathcal{T}_j}([x]) \) is the term \( [x^j] \) of a sequence \( [x^{k+1}] = C_{\mathcal{T}_j}([x^j]) \), under some stopping criteria, and \( C_{\mathcal{T}_j}([x]) \) is the contractor (5-9) with a disturbance input \( d^i \) instead of \( d \). The samples \( d^i \) are those of the set \( \mathcal{D} \) defined on (5-8).

The contraction procedure for the outer bound begins just like on the elimination procedure by integrating the box forward and checking which boxes on the target set it intersects in the end, \( [\varphi](r, [x], [\mathbb{U}], d^i) \cap \mathcal{T} \).

As was done for the B&P algorithm, an associated list was created for every box on the subpaving and every sampled disturbance input \( d^i \in \mathcal{D} \). Also on \( C_{\mathcal{T}}([x]) \), when computing \( [\varphi](r, [x], [\mathbb{U}], d^i) \cap \mathcal{T} \) the associated list of \( C_{\mathcal{T}}([x]) \) and \( d^i \) stores a pointer for every box on \( \mathcal{T} \) that intersects with \( [\varphi](r, [x], [\mathbb{U}], d^i) \), and when bisecting a box its children inherit its associated lists.
After this first step, \([\varphi](r, [x], [u], d) \cap T\), if we take the interval union of the boxes on the target it intersected and then intersect them with the integrated box we obtain an enclosure of the states on the target set the system can reach from the original box, \([[\varphi](r, [x], [u], d) \cap T\]). Integrating that enclosure backwards again and intersecting the result with the original box, \([x] \cap [\varphi_{inv}](r, [[\varphi](r, [x], [u], d) \cap T], [u], d)\), we obtain an enclosure of the initial states on the original box from which the target set may be reached, all the other regions of the box may be eliminated as was stated on assumption III.

An algorithm was devised that uses the function \(B&P(test(\cdot), [x]_0, \varepsilon, T)\), or \(B&C(C_T(\cdot), [x]_0, \varepsilon, T)\) to compute a subpaving containing a reachable set of a dynamical system. That algorithm will be named SubReach and will admit as inputs the root of the subpaving \([x]_0, \varepsilon\), the list \(T\), the bounds on the control input \(\mathbb{U}\), the bounds on the disturbance input \(\mathbb{D}\), an over-approximation of the target set \(T_\varepsilon^+\), the final time \(t_f\), the number of samples on each dimension \(n_s\), and the number of time steps \(N_{stp}\). Also, there are parameters for the guaranteed integration solvers that must be defined. The algorithm is as follows.

**Algorithm: SubReach** (out: \(T^+(0)\), in: \([x]_0, \varepsilon, T, \mathbb{U}, \mathbb{D}, T^+_\varepsilon, t_f, n_s, N_{stp}\))

1. \(\tau := t_f/N_{stp}\)
2. Compute the sampling set for the disturbance input \(\mathbb{D}\) as on (5-8) using \(n_s\) and \(\mathbb{D}\).
3. \(T := T^+_\varepsilon\)
4. \(j := 1\)
5. Do
6. \(T = B&P(test(\cdot), [x]_0, \varepsilon, T)\) or \(T = B&C(C_T(\cdot), [x]_0, \varepsilon, T)\)
7. While \(j \leq N_{stp}\)
8. \(T^+(0) := T\)

On step 6 of the previous algorithm \(test(\cdot)\) or \(C_T(\cdot)\) use as parameters \(\tau, \mathbb{U}, \mathbb{D}\), the subpaving \(T\) on the last time step, among other parameters used by the guaranteed integration solver that will be discussed further ahead. One possibility of effectively defining the set \(T\) is as the set containing the bounds of the box \([x]_0\). Although other possibilities may be considered, on the rest of this thesis, when computing outer bounds we will only consider that definition of the set \(T\).

### 5.6 Inner bound

The application of the B&P and B&C algorithms to compute inner bounds is based on Assumptions I, II and III. From Assumption I we have that on the test function and contractors used to compute the inner bound instead of using an enclosure of the admissible control inputs and samples of admissible disturbance inputs, we use samples of admissible control inputs and an enclosure of the admissible disturbance inputs. On that basis, the test function and contractor used to compute inner bounds of reachable sets are the following:

\[
    test([x]) = \begin{cases} 
        1, & \text{if } [\varphi](r, [x], u, [\mathbb{D}]) \cap T \neq \emptyset \\
        0, & \text{if } [\varphi](r, [x], u, [\mathbb{D}]) \cap T = \emptyset 
    \end{cases} \tag{5-10}
\]
For computing the inner bound the intermediate sets $\mathcal{T}$ are overestimations of the complementary set of the reachable set at intermediate times, $\mathcal{T} = \mathcal{T}^{c+}(\tau) \cap [x]_0$. The rationale for this method is that if we integrate a box forward and intersect the result with an overestimation of the complement of the target set, $[\varphi](\tau, [x], u, [\mathcal{D}]) \cap \mathcal{T}$, we obtain an enclosure of the states outside the target set that may be reached under that particular control.

If we integrate that enclosure backwards and intersect the result with the original box, $[x] \cap [\varphi^{-1}](\tau, [\varphi](\tau, [x], u, [\mathcal{D}]) \cap \mathcal{T}, u, [\mathcal{D}])$, we obtain an enclosure of the initial sets from where the target set is not reached under that particular control, all the other parts of the box are guaranteed to belong to the reachable set, as is stated on assumption III.

As referred, Instead of considering the control as an interval number, a number of samples of the control are taken and tested. The sampling set is defined as follows.

$$\mathcal{U} = \left\{ d \in \mathbb{R}^m; \; d_i = \frac{\uparrow u_i^{j-1} + \downarrow u_i^{ns-j}}{ns-1}, i \in \{1, \ldots, m\}, j \in \{1, \ldots, ns\} \right\}$$

The process of computing the inner bound is not much different from the outer bound. The initial target set considered is the complementary set of the real target set. The devised algorithm for computing inner bounds is the following:

**Algorithm: SubReach (out: $T^-(0)$, in: $[x]_0, \varepsilon, J, \mathcal{D}, \mathcal{T}_0^{c+}, t_f, ns, Nstp$)**

1. $\tau := t_f/Nstp$
2. Compute the sampling set for the disturbance input $\mathcal{D}$ as on (5-12) using $ns$ and $\mathcal{U}$
3. $\mathcal{T} := \mathcal{T}_0^{c+}$
4. $j := 1$
5. Do
6. \[ \mathcal{T} = \text{B&P}(\text{test}(\cdot), [x]_0, \varepsilon, J) \text{ or } \mathcal{T} = \text{B&C}(C_{\mathcal{T}}(\cdot), [x]_0, \varepsilon, J) \]
7. Insert on the subpaving $\mathcal{T}$ the boxes resulting from \textit{exclude} ([x]_0, \mathbb{R}^n)
8. While $j \leq Nstp$
9. \( \mathcal{T}^-(0) := \mathcal{T}^c \)

On step 6 of the previous algorithm \textit{test}(\cdot) or $C_{\mathcal{T}}(\cdot)$ use as parameters $\tau, \mathcal{U}, \mathcal{D}$, the subpaving $\mathcal{T}$ on the last time step, among other parameters used by the guaranteed integration solver that will be discussed further ahead. For this algorithm an effective form of initializing the set $\mathcal{T}$ is defining it as containing boxes known to belong to the target set $\mathcal{T}_0$.

### 5.7 Applying the Euler Method

To compute the guaranteed integration of the flow one possible method is the Euler method. On this thesis the Euler method is implemented as described on the first section of chapter 4. The method
described uses as parameters the expansion factors $\alpha$ and $\beta$. Those parameters must be chosen carefully since selecting high values leads to a large overestimation, while choosing low values may lead to failure of the method ($[q](t, [x]) = \mathbb{R}^n$).

To reduce the overestimation it is preferable to compute the enclosure of the integration in $N_{stp_{integ}}$ time steps. For that purpose, naming the Euler method as described on the first section of chapter 4 as $[q]_{step}$, the application of the Euler method to compute $[q](t, [x], [U, D])$ on the rest of this work will be performed as follows.

**Algorithm** $[q]$ (inout: $[x]$, $[U, D]$, $t_{end}$, $N_{stp_{integ}}$)

1. Leave := 0 ; $t_{step} := t_{end} / N_{stp_{integ}}$; $t := t_{step}$
2. $[x] := [q]_{step}(t, [x], [U, D])$
3. if $t = t_{end}$ return;
4. $t := t + t_{step}$ ; goto 2 ;

5.8 Applying Interval Taylor Models

To apply the guaranteed integration using Interval Taylor Models the possibility considered in this work is using the VSPODE solver.

To verify the potentialities of the VSPODE solver, it is possible to retrieve the result as an interval Taylor model. This Taylor model is a function of virtual coordinates with the same dimension as the unknown quantities in the initial value problem.

A program was created which plots the results given in VSPODE in two dimensions and not including the remainder part of the Taylor model.

The approach is using a vector of class Taylor instead of a vector of class Interval. A vector of class Taylor contains for each element the coefficients of the Taylor model and the remainder part. The box enclosing the virtual coordinates used on the Taylor models can also be obtained.

Passing the information obtained as described above to MATLAB™ using a mex-file, the borders of the nonlinear set are plotted. This function plots the lines obtained computing the polynomial part of the Taylor model on the sides of the box of virtual coordinates.

This approach was tested for the following system:

$$\begin{align*}
\frac{dx_1}{dt} &= x_2 \\
\frac{dx_2}{dt} &= x_1^2
\end{align*}$$

(5-13)

The obtained results are displayed in the following figure.
VSPODE was applied to the Branch and Prune and Branch and Contract algorithms described before.

The VSPODE solver admits the following parameters:

- The order of the Taylor expansion in time, which regulates the order of the Taylor series approximation of the flow function.

- The order of the Taylor model, which regulates the order of the Taylor model enclosing the result and the intermediate steps.

- The time step.

This completes the discussion of the computational implementation of the proposed method. In the next chapter the results obtained are presented.

5.9 Conclusions

In this chapter some algorithms of computing subpavings alternative to SIVIA of section 3.6 were described. The main feature of the described methods is that only boxes in contact with already eliminated boxes or with a predefined set of boxes \( J \) can be eliminated or contracted.

It was concluded that the definition of set \( J \) has effects on both the efficiency and effectiveness of the method. If set \( J \) is very conservative, i.e. covers more regions than necessary, some boxes on the set to be enclosed may be evaluated and bisected unnecessarily thus compromising the efficiency and if set \( J \) covers less regions than necessary there may be regions that do not belong to the set and are enclosed by the resulting subpaving, thus compromising the effectiveness.

It was also observed that all the devised methods of computing subpavings suffer from the “curse of dimensionality”, that is, the time to run grows exponentially with the dimension of the problem.

We have also seen how to apply the Euler method for guaranteed integration computationally. A possible method to improve the quality of the results is to perform the integration in multiple time steps as was described in section 5.7.
Finally, we have concluded that it is possible to integrate a guaranteed integration solver programmed on C++ such as VSPODE, with the developed methods programmed on MATLAB™, by using mex-files. It is also possible to retrieve to MATLAB™ the polynomial coefficients of the Taylor Model obtained with VSPODE, which allows to visualize the shape of the result.

This completes the discussion of the computational implementation of the proposed method. On the next chapter the results obtained with the devised methods are presented.

Chapter 6: Application Example

To verify the performance of the developed methods, the methods were applied in a Matlab™ program and were used to solve the double integrator system (6.1), with \( u \in [-1,1] \) and a final time \( t_f = 1 \).

\[
\dot{x} = \begin{bmatrix} \dot{x}_2 \\ u \end{bmatrix} = f(x,u)
\]  \tag{6.1}

The first section of this chapter, section 6.1, contains the analytical computation of reachable sets of the double integrator system. The following section, 6.2, shows the results of the application of the Branch and Prune algorithm with flag from last chapter to compute inner and outer bounds of Reachable sets. Section 6.3 contains the results from the application of the branch and contract algorithms to compute inner and outer bounds of reachable sets. Section 6.4 holds the results obtained when using VSPODE to perform the guaranteed integration. The final section, 6.5, contains the conclusions from this chapter.

6.1 Analytical Solution

The analytical solution for the double integrator system can be computed using the fact that the optimal control is of type “bang-bang”, since the system function is linear in the control variable. The optimal control is of the following type, where \( t_{trans} \in [0,1] \):

\[
u(t) = \begin{cases} 
1, & \text{if } 0 \leq t \leq t_{trans} \\
-1, & \text{if } t_{trans} < t \leq 1
\end{cases} \quad \text{or} \quad u(t) = \begin{cases} 
-1, & \text{if } 0 \leq t \leq t_{trans} \\
1, & \text{if } t_{trans} < t \leq 1
\end{cases}
\]

To compute the solutions \((x_1(0), x_2(0))\) on the boundary of the backwards reachable set \( T(0) \) we have to integrate the system subject to an optimal control. In the case we use the control \( u(t) = \begin{cases} 
1, & \text{if } 0 \leq t \leq t_{trans} \\
-1, & \text{if } t_{trans} < t \leq 1
\end{cases} \) we obtain the following solution:

\[
x_2(t) = x_2(t_f) - \int_t^{t_f} u(s) \, ds = x_2(t_f) + \begin{cases} 
- \int_t^{t_{trans}} (-1) \, ds - \int_t^{t_{trans}} 1 \, ds, & \text{if } 0 \leq t \leq t_{trans} \\
- \int_t^{t_{trans}} (-1) \, ds, & \text{if } t_{trans} < t \leq 1
\end{cases}
\]

\[
= x_2(t_f) + \begin{cases} 
t + (1 - 2t_{trans}), & \text{if } 0 \leq t \leq t_{trans} \\
(1 - t), & \text{if } t_{trans} < t \leq 1
\end{cases}
\]

71
\[ x_1(t) = x_1(t_f) - \int_t^{t_f} x_2(s) \, ds \]
\[ = x_1(t_f) - \int_t^1 x_2(t_f) \, dt \]
\[ + \begin{cases} - \int_1^{t_{trans}} (1-s) \, ds - \int_t^{t_{trans}} s + (1-2t_{trans}) \, ds, & \text{if } 0 \leq t \leq t_{trans} \\ - \int_t^1 (1-s) \, ds, & \text{if } t_{trans} < t \leq 1 \end{cases} \]
\[ = x_1(t_f) - (1-t) x_2(t_f) + \begin{cases} \frac{1}{2} t^2 + (1-2t_{trans})t + t_{trans}^2 - \frac{1}{2}, & \text{if } 0 \leq t \leq t_{trans} \\ -\frac{1}{2} t^2 + t - \frac{1}{2}, & \text{if } t_{trans} < t \leq 1 \end{cases} \]

Or using the other control form \( u(t) = \begin{cases} -1, & \text{if } 0 \leq t \leq t_{trans} \\ 1, & \text{if } t_{trans} < t \leq 1 \end{cases} \) we have:

\[ x_2(t) = x_2(t_f) - \begin{cases} t + (1-2t_{trans}), & \text{if } 0 \leq t \leq t_{trans} \\ (1-t), & \text{if } t_{trans} < t \leq 1 \end{cases} \]
\[ x_1(t) = x_1(t_f) + (1-t) x_2(t_f) - \begin{cases} \frac{1}{2} t^2 + (1-2t_{trans})t + t_{trans}^2 - \frac{1}{2}, & \text{if } 0 \leq t \leq t_{trans} \\ -\frac{1}{2} t^2 + t - \frac{1}{2}, & \text{if } t_{trans} < t \leq 1 \end{cases} \]

The solutions on the boundary of the backward reachable sets \((x_1(0), x_2(0))\) are then expressed by \((x_1(t_f), x_2(t_f)) + (t_{trans}^2 - \frac{1}{2} - x_2(t_f), (1-2t_{trans}))\) or \((x_1(t_f), x_2(t_f)) - (t_{trans}^2 - \frac{1}{2} - x_2(t_f), (1-2t_{trans}))\) with \(t_{trans} \in [0,1]\).

For the target set \(T_0 = (0,0)\) the states on the boundary are expressed by \((t_{trans}^2 - \frac{1}{2}, (1-2t_{trans}))\) or by \((\frac{1}{2} - t_{trans}^2, -(1-2t_{trans}))\) with \(t_{trans} \in [0,1]\). The analytical solution for \(T_0 = (0,0)\) is displayed on the following figure.

![Figure 6-1: Analytical solution for target set (0,0)](image-url)
When the target set is $\mathcal{T}_0 = ([0.2,0.2], [-0.2,0.2])$, the boundary of the backward reachable set is composed by six segments. We have the boundaries originated from the sides of the target set $(0.2,[-0.2,0.2])$ and $([-0.2,0.2],0.2)$ with the optimal control $u(t) = -1$ that can be described by $(-s + 0.3), (s + 1)$ and $(s - 0.7,1.2)$, with $s \in [-0.2,0.2]$. We also have the boundaries originated from the sides $(-0.2,[-0.2,0.2])$ and $([-0.2,0.2],-0.2)$ with the optimal control $u(t) = 1$ that are described by $(s + 0.3, -(s + 1))$ and $(0.7 - s, -1.2)$, with $s \in [-0.2,0.2]$. There is also the segment originating from the point of the target set $(-0.2,0.2)$, with controls of the form $u(t) = \begin{cases} 1, & \text{if } 0 \leq t \leq t_{\text{trans}} \\ -1, & \text{if } t_{\text{trans}} < t \leq 1 \end{cases}$, described by $(-0.9 + t_{\text{trans}}^2, 1.2 - 2t_{\text{trans}})$, with $t_{\text{trans}} \in [0,1]$. Finally we have the segment originating from the point of the target set $(0.2,-0.2)$, with controls of the form $u(t) = \begin{cases} -1, & \text{if } 0 \leq t \leq t_{\text{trans}} \\ 1, & \text{if } t_{\text{trans}} < t \leq 1 \end{cases}$, described by $(0.9 - t_{\text{trans}}^2, -1.2 + 2t_{\text{trans}})$, with $t_{\text{trans}} \in [0,1]$. The analytical solution for $\mathcal{T}_0 = ([0.2,0.2], [-0.2,0.2])$ is displayed on Figure 6-2.

**Figure 6-2: Analytical solution for target set ([0.2,0.2], [-0.2,0.2])**

### 6.2 Branch and Prune

First we present the results of the Branch and Prune algorithm using the test function (5-7) for the outer bound and (5-10) for the inner bound.

The result of $\mathcal{T}^+(0)$, for $\mathcal{T}_0 = (0,0)$, computed with one time step, with a minimum box size of 0.05 and an initial box $[x]_0 = ([0.2,0.2], [-0.2,0.2])$ is displayed in the following figure.
Figure 6-3: Over-approximation of reachable set computed with one time step

It is observable that there is a great deal of overestimation. This is due to the effect of the overestimation due to the guaranteed overestimation, $o e_{\text{encl}}(t/Nstp,[\varphi],...)$, defined on (4-15), mainly because of the wrapping effect discussed earlier.

As discussed before, a solution to this problem is increasing the number of time steps reducing the effect of $o e_{\text{encl}}(t/Nstp,[\varphi],...)$.

The overapproximation of the backward reachable set $\mathcal{T}^+(0)$, for $T_0 = (0,0)$, computed with 10 time steps, a minimum box size of 0.005 and an initial box $[x]_0 = ([-2.2],[-2.2])$ is displayed on Figure 6-4. The integration is done by means of the Euler method described on the Previous chapter with $\alpha = 0.01, \beta = 0.001$ and $Nstp_{\text{integ}} = 16$.

Figure 6-4: Over-approximation of reachable set computed with 10 time steps

From the last example it is observable that using several time steps contributes to reducing the overestimation, as was expected.
To compute the inner bound it is necessary that the target set $T_0$ has a relatively large width or else it is impossible that the test (5-10) is true for some box. On the following examples when constructing the set $T$ defined on (5-12) we will use $ns = 2$, that is, for each time step we consider the controls $u = -1$ and $u = 1$.

The set $T^{fc}(0) \cap [x]_0$, for $T_0 = ([0,0.2],[0,0.2])$, computed with 10 time steps, with a minimum box size of 0.005 and an initial box $[x]_0 = ([2,2],[2,2])$ is shown on Figure 6-5. The guaranteed integration is done through the Euler method with $\alpha = 0.001$, $\beta = 0.0001$ and $Nstp_{integ} = 16$.

![Figure 6-5: Under-approximation of reachable set computed with 10 time steps](image)

On the last example there is a clear underestimation due to the sampling the control only with $u = -1$ and $u = 1$, which is expressed by $oe_{assume}$ defined on (4-14). The controls that result on the inner bound computed on Figure 6-5 have the same form of the analytical solution of the last section where $t_{trans} \in [0,1]$, while on the optimal control we would have $t_{trans} \in [0,1]$, thus the observable "saw-tooth" effect on the figure above.

To obtain a more general picture, the results of $cl(T^{fc}(0) \cap [x]_0)$ and $cl(T^{-c}(0))$, for $T_0 = ([0,0.2],[0,0.2])$, for 10 time steps, a minimum box size of 0.005 and with $[x]_0 = ([2,2],[2,2])$ are displayed on Figure 6-6. The area of $T^c(0)$ computed with the sum of the area (Vol([x])) of all boxes of the subpaving describing $T^c(0)$ is 2.3779 and the area of $T^{fc}(0) \cap [x]_0$ computed with the same method is 14.4002. The guaranteed integration is done with the Euler method with parameters $\alpha = 0.001$, $\beta = 0.0001$ and $Nstp_{integ} = 16$. 
Figure 6-6: Inner and outer bound of reachable set for the double integrator example using the B&P algorithm

The blue line on the last figure represents the boundary of the true reachable set. It is observable that there is still too much over and underestimation. It is expected that by using a branch and contract algorithm some of the overestimation will be eliminated.

6.3 Branch and Contract

On this section contains the results obtained when using the B&C algorithm. The results displayed on this section are for $cl(T^+(0) \cap [x]_0)$ and $cl(T^-(0))$, with a target set $T_0 = ([0,0.2],[0,0.2])$ and with a minimum box size of 0.005. The initial box, root of the subpaving is $[x]_0 = ([2,2],[2,2])$.

The computed set $T^+(0)$ obtained with one time step is displayed in Figure 6-7. The area of the computed set $T^+(0)$ is 2.3869. The Euler method is used for guaranteed integration with $\alpha = 0.001$, $\beta = 0.0001$ and $Nstp_{integ} = 256$. 

Figure 6-7: Double integrator example with one time step using contractors
Since the contractor (5-9) involves performing integrations forward and backward there is less overestimation due to the guaranteed integration, specially on the $x_1$ direction.

The sets $cl(T^+(0) \cap [x]_0)$ and $cl(T^-(0))$ computed with 10 time steps are on Figure 6-8. The area of $T^+(0)$ computed with 10 time steps is 2.1005 and the area of $T^-(0) \cap [x]_0$ is 14.2369. The guaranteed integration is done by means of the Euler method with $\alpha = 0.001$, $\beta = 0.0001$ and $N_{stp_{integ}} = 16$.

![Figure 6-8: Double integrator example with 10 time steps using contractors](image)

From Figure 6-8 and from the area of the resulting subpavings we can conclude that there is a big advantage in using a branch and contract algorithm when comparing to a branch and prune algorithm.

The results $cl(T^+(0) \cap [x]_0)$ and $cl(T^-(0))$ for 20 time steps are displayed in Figure 6-9. The area of $T^+(0)$ is 2.1748 and of $T^-(0) \cap [x]_0$ is 14.1549. The parameters used on the Euler method for the guaranteed integration are $\alpha = 0.001$, $\beta = 0.0001$ and $N_{stp_{integ}} = 16$. 
Figure 6-9: Double integrator example with 20 time steps using contractors

There is an expected reduction on the underestimation of the inner bound due to considering on the sampled control described above \( t_{\text{trans}} \in \{0, 0.05, 0.1, \ldots, 1\} \). As for the outer bound there is a slight increasing in the overestimation, due to the wrapping effect at intermediate time steps described by \( o e_{sp}(\varepsilon) \).

6.4 VSPODE

To observe the effects of using VSPODE to perform the guaranteed integration, the results of \( cl(T^+(0) \cap [x]_0) \) and \( cl(T^-(0)) \) using VSPODE for state propagation, for 10 time steps with a minimum box size of 0.005 are displayed in Figure 6-10. The target set is \( T_0 = ([0, 0.2], [-0.2, 0.2]) \) and the initial box is \([x]_0 = ([0, 2], [-0.2, 0.2]) \). The parameters used by VSPODE are a time step of 0.0001 an order of the Taylor model 15 and an order of the Taylor expansion in time 7.
From the results of Figure 6-10 the computed area of $\mathcal{T}^+(0)$ is 2.2087 and the area of $\mathcal{T}^{-\epsilon}(0) \cap [x]_0$ is 14.2640.

We conclude that the over and underestimation using VSPODE is greater than using the Euler method which indicates that there is a necessity of future work in this area.

6.5 Conclusions

On this chapter we have concluded that the devised methods are applicable and provide meaningful results. When implementing the method with the B&P algorithm we observed that using multiple time steps may reduce the overestimation of the result.

To better illustrate the results obtained on this chapter, Table 6-1 and Table 6-2 show the relative overestimation and underestimation of the covered areas (the area of the analytical solution is 13.973). The overestimation and underestimation are computed from equations (6-1) and (6-2).

\[
\text{Overestimation} = \frac{(\text{covered area} - \text{analytical sol. area})}{\text{analytical sol. area}} \quad (6-1)
\]

\[
\text{Underestimation} = \frac{(\text{analytical sol. area} - \text{covered area})}{\text{analytical sol. area}} \quad (6-2)
\]
<table>
<thead>
<tr>
<th>Method (10 Time steps)</th>
<th>Overestimation</th>
<th>Underestimation</th>
</tr>
</thead>
<tbody>
<tr>
<td>B&amp;P – Euler Method</td>
<td>17.3%</td>
<td>21.1%</td>
</tr>
<tr>
<td>B&amp;C – Euler Method</td>
<td>3.6%</td>
<td>13.0%</td>
</tr>
<tr>
<td>B&amp;C – VSPODE</td>
<td>8.9%</td>
<td>14.3%</td>
</tr>
</tbody>
</table>

Table 6-1: Overestimation and Underestimation of different methods

<table>
<thead>
<tr>
<th>Time steps</th>
<th>Overestimation</th>
<th>Underestimation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>17.8%</td>
<td>100%</td>
</tr>
<tr>
<td>10</td>
<td>3.6%</td>
<td>13.0%</td>
</tr>
<tr>
<td>20</td>
<td>7.3%</td>
<td>8.9%</td>
</tr>
</tbody>
</table>

Table 6-2: Overestimation and Underestimation of different numbers of time steps

When employing the B&C algorithm we observe that the overestimation is much lower than that obtained when using the B&P algorithm. The results for the outer bound obtained with 20 time steps are worse than those obtained with 10 time steps. This confirms the theory of section 4.2 that says that there is a number of time steps $N_{stp}$ that provides a minimum of overestimation and for the outer bound on the B&C algorithm that number is between 1 and 20.

Finally, when applying VSPODE to perform the guaranteed integration part we obtain worse results than when applying the Euler method. This fact indicates that further work is necessary since when using Taylor models the result should be better, or at least have the same quality, than when using boxes to enclose the intermediate results.
Chapter 7: Conclusions and recommendations

On this chapter we will review the main conclusions and recommendations that can be derived from this work both at a theoretical level and from the point of view from the computational application.

On section 7.1 the main focus will be to summarize the work done and the findings during the course of the work. On section 7.2 a number of suggestions for future work are outlined and briefly argued.

7.1 Conclusions

From the theory review we can conclude that there is a lot of work done in the fields of interval analysis and viability theory (reachable sets). However, until very recently we have seen little work on the application of interval analysis to compute guaranteed bounds on reachable sets of continuous systems. This work gives some contribution on this matter.

We have seen that using the field of interval analysis, namely the construction of subpavings through set inversion and guaranteed integration, we can compute inner and outer bounds of reachable sets defined by the union of non-overlapping boxes.

An algorithm of constructing subpavings (B&P) was implemented where only boxes in contact with already eliminated boxes or in contact with a pre-defined set of boxes \( J \) can be eliminated. This property is useful to avoid evaluating boxes unnecessarily but can lead to an additional overestimation if the predefined set of boxes \( J \) is poorly defined. The algorithm was tested and it was proven to work as expected, however suffering from the “curse of dimensionality”, the time to run grows exponentially with the dimension of the problem.

Because on the problem at hand the test function or the contractor does not have parameters that change throughout the algorithm we can modify the B&P algorithm to be more efficient by preventing that boxes that have already been evaluated be evaluated again (B&P algorithm with flag).

With a small alteration from the B&P algorithm with flag we can use contractors to construct subpavings instead of test functions (B&C), yielding a tighter result with fewer iterations. The B&C algorithm has different properties from the B&P algorithm, for example it is not always necessary that if the predefined set of boxes \( J \) is poorly defined that an additional overestimation will occur.

From the results of the devised method applied to compute bounds on the reachable set of the double integrator system we can draw some conclusions. We observe that using a branch and contract algorithm yield much tighter results than using a branch and prune algorithm. Increasing the number of time steps above a certain number may have a detrimental effect on the outer bound since at each time step we have a wrapping effect of the intermediate solution, but has a positive effect on the inner bound since we are considering more samples of the control input. The application of VSPODE showed that it yields worse results than Euler integration indicating that there is still work to be done either on how the program is applied or on VSPODE itself.
7.2 Recommendations

As a recommendation of future work, to devise new guaranteed integration methods and apply them on the methods discussed on this work or to apply other available guaranteed integration programs may improve the effectiveness of the discussed methods. Also, the application of VSPODE on the created programs may be improved or possibly another program that implements interval Taylor models may be created since it looks that they can have a great advantage comparing to use interval numbers only.

It was observed that programming on C++ yielded much more computationally efficient programs than those implemented on MATLAB™ the applicability of the devised methods may improve significantly if the methods were programmed on C++.

Finally, another possible future work is, instead of using samples of the control input or disturbance input, we can estimate the optimal control input or disturbance input knowing that the optimal control input is the input that maximizes the system function on the direction towards the reachable set at a particular moment and state, and the optimal disturbance input is the input that minimizes the state function towards the reachable set at that moment.
Bibliography


Appendices

Appendix A: mean value evaluation

Extract of Nedialkov’s SWIM 2008 conference slides on guaranteed integration (Nedialkov N. , 2008).

Reducing the wrapping effect

On each step, represent the enclosure in the form

\[ y_j \in \{ \tilde{y}_j + A_j r_j \mid r_j \in R^n \} \], \quad A_j \in R^{n \times n} \text{ nonsingular}

Instead of computing

\[ y_{j+1} = y_j + S_j(y_j - \tilde{y}_j), \]

compute

\[ y_{j+1} = y_j + S_j(y_j - \tilde{y}_j), \]

where \( r_0 = y_0 - \tilde{y}_0 \), \( A_0 = I \), \( m(\cdot) \) is midpoint

How to select \( A_{j+1} \)?

The Parallelepiped Method

\[ A_{j+1} = m(S_j(A_j)) \]

The \( A_j \) usually become ill conditioned

Lohner’s QR Method

\[ A_{j+1} = Q_{j+1} \]

from the QR factorization \( Q_{j+1} R_{j+1} = m(S_j A_j) \)

We enclose in a moving orthogonal coordinate system

We can always “match” the longest edge of the enclosed set

The QR method provides better stability than the parallelepiped method (NN, K. Jackson)
Appendix B: Tables with data from algorithms

Time to run (seconds) as a function of the minimum box size and number of dimensions for the B&P algorithm:

<table>
<thead>
<tr>
<th>A</th>
<th>Number of dimensions</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0.029</td>
<td>0.084</td>
<td>0.301</td>
<td>0.98</td>
<td>2.77</td>
<td>7.672</td>
<td>20.703</td>
<td>55.86</td>
<td>152.466</td>
<td>433.071</td>
</tr>
<tr>
<td>0.5</td>
<td>0.039</td>
<td>0.157</td>
<td>0.93</td>
<td>7.204</td>
<td>45.66</td>
<td>748.368</td>
<td>2717.952</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.25</td>
<td>0.051</td>
<td>0.574</td>
<td>9.646</td>
<td>194.456</td>
<td>4182.468</td>
<td>8651.53</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.125</td>
<td>0.062</td>
<td>1.689</td>
<td>104.409</td>
<td>13507.01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0625</td>
<td>0.083</td>
<td>5.913</td>
<td>2012.349</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.03125</td>
<td>0.082</td>
<td>27.422</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.015625</td>
<td>0.091</td>
<td>114.202</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.007813</td>
<td>0.102</td>
<td>615.518</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.003906</td>
<td>0.121</td>
<td>2408.628</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.001953</td>
<td>0.143</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.000977</td>
<td>0.162</td>
<td>615.518</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.000488</td>
<td>0.186</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.000244</td>
<td>0.217</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.000122</td>
<td>0.233</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.3E-05</td>
<td>0.268</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.05E-05</td>
<td>0.3</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.53E-05</td>
<td>0.337</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.63E-06</td>
<td>0.402</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.81E-06</td>
<td>0.415</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.91E-06</td>
<td>0.448</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.54E-07</td>
<td>0.475</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.77E-07</td>
<td>0.508</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.38E-07</td>
<td>0.552</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.19E-07</td>
<td>0.608</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.96E-08</td>
<td>0.66</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.98E-08</td>
<td>0.703</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.49E-08</td>
<td>0.749</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.45E-09</td>
<td>0.804</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.73E-09</td>
<td>0.86</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.86E-09</td>
<td>0.927</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.31E-10</td>
<td>0.973</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.66E-10</td>
<td>1.032</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.33E-10</td>
<td>1.117</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.16E-10</td>
<td>1.154</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.82E-11</td>
<td>1.249</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.91E-11</td>
<td>1.286</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.46E-11</td>
<td>1.362</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.28E-12</td>
<td>1.416</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.64E-12</td>
<td>1.491</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.82E-12</td>
<td>1.562</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.09E-13</td>
<td>1.671</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.55E-13</td>
<td>1.756</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.27E-13</td>
<td>1.876</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.14E-13</td>
<td>1.928</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.68E-14</td>
<td>2.049</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.84E-14</td>
<td>2.09</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.42E-14</td>
<td>2.161</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.11E-15</td>
<td>2.241</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.55E-15</td>
<td>2.367</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.78E-15</td>
<td>2.498</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.88E-16</td>
<td>2.551</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.44E-16</td>
<td>2.639</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.22E-16</td>
<td>2.73</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.11E-16</td>
<td>2.838</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.55E-17</td>
<td>2.945</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.27E-17</td>
<td>3.052</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.14E-17</td>
<td>3.159</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.68E-18</td>
<td>3.266</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.84E-18</td>
<td>3.373</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.42E-18</td>
<td>3.480</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.11E-19</td>
<td>3.587</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.55E-19</td>
<td>3.694</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.78E-19</td>
<td>3.801</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.88E-20</td>
<td>3.908</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.44E-20</td>
<td>4.015</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.22E-20</td>
<td>4.122</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.11E-20</td>
<td>4.229</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.55E-22</td>
<td>4.336</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.78E-22</td>
<td>4.443</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.39E-22</td>
<td>4.550</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.94E-24</td>
<td>4.657</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.47E-24</td>
<td>4.764</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.73E-24</td>
<td>4.871</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.66E-26</td>
<td>4.978</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.33E-26</td>
<td>5.085</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.17E-26</td>
<td>5.192</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.08E-28</td>
<td>5.299</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.40E-29</td>
<td>5.406</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.70E-29</td>
<td>5.513</td>
<td>10021.425</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Number of boxes on the result as a function of the minimum box size and number of dimensions (left) and processed boxes during the B&P algorithm as a function of the minimum box size and number of dimensions (right):

<table>
<thead>
<tr>
<th>A</th>
<th>nd</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.5</td>
<td>4</td>
<td>16</td>
<td>64</td>
<td>240</td>
<td>832</td>
<td>2688</td>
<td>8192</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.25</td>
<td>6</td>
<td>44</td>
<td>316</td>
<td>2336</td>
<td>13952</td>
<td>73984</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.125</td>
<td>8</td>
<td>104</td>
<td>1528</td>
<td>20664</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.0625</td>
<td>10</td>
<td>228</td>
<td>65536</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0.03125</td>
<td>12</td>
<td>472</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0.015625</td>
<td>14</td>
<td>940</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0.0078125</td>
<td>16</td>
<td>1888</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0.00390625</td>
<td>18</td>
<td>3776</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>0.001953125</td>
<td>20</td>
<td>7552</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0.0009765625</td>
<td>22</td>
<td>15104</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>0.00048828125</td>
<td>24</td>
<td>30208</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>0.000244140625</td>
<td>26</td>
<td>60416</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>0.0001220703125</td>
<td>28</td>
<td>120832</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>0.00006103515625</td>
<td>30</td>
<td>241664</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>0.000030517578125</td>
<td>32</td>
<td>483312</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>0.0000152587890625</td>
<td>34</td>
<td>966624</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>0.00000762939453125</td>
<td>36</td>
<td>1933248</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>0.000003814697265625</td>
<td>38</td>
<td>3866496</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>0.0000019073486328125</td>
<td>40</td>
<td>7732992</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>0.00000095367431640625</td>
<td>42</td>
<td>15465984</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>0.000000476837208125</td>
<td>44</td>
<td>30931968</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>0.00000023841860390625</td>
<td>46</td>
<td>61863936</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>0.000000119209301953125</td>
<td>48</td>
<td>123727872</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>0.0000000596046509765625</td>
<td>50</td>
<td>247455744</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>0.00000002980232548775390625</td>
<td>52</td>
<td>494911488</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>0.00000001490116274387796875</td>
<td>54</td>
<td>989822976</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>0.000000007450581371938984375</td>
<td>56</td>
<td>1979645952</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>0.000000003725290685969492188</td>
<td>58</td>
<td>3959291904</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>0.000000001862645342984746093</td>
<td>60</td>
<td>7918583808</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>0.000000000931322671492373047</td>
<td>62</td>
<td>15837167616</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>0.000000000465661335746186523</td>
<td>64</td>
<td>31674335232</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>0.000000000232830667873093262</td>
<td>66</td>
<td>63348670464</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>0.000000000116415333936546631</td>
<td>68</td>
<td>126697340928</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>0.000000000058207666968273316</td>
<td>70</td>
<td>253394681856</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>0.000000000029103833484136658</td>
<td>72</td>
<td>506789363712</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>0.000000000014551916742068329</td>
<td>74</td>
<td>1013578727424</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>0.000000000007275958371034164</td>
<td>76</td>
<td>2027157454848</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>0.000000000003637979185517082</td>
<td>78</td>
<td>4054314909696</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>0.000000000001818989592778541</td>
<td>80</td>
<td>8108629819392</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>0.000000000000909494796389271</td>
<td>82</td>
<td>16217259638784</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>0.000000000000454747398194635</td>
<td>84</td>
<td>32434519277568</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>0.000000000000227373699097318</td>
<td>86</td>
<td>64869038555136</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>43</td>
<td>0.000000000000113686849548659</td>
<td>88</td>
<td>129738077110272</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>44</td>
<td>0.000000000000056843424774329</td>
<td>90</td>
<td>259476154220544</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>45</td>
<td>0.000000000000028421712387165</td>
<td>92</td>
<td>518952308441088</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>46</td>
<td>0.000000000000014210856193582</td>
<td>94</td>
<td>1037904616882176</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>47</td>
<td>0.000000000000007105428096790</td>
<td>96</td>
<td>2075809233764352</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>48</td>
<td>0.000000000000003552714048395</td>
<td>98</td>
<td>4151618467528704</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>49</td>
<td>0.000000000000001776357024198</td>
<td>100</td>
<td>8303236935057408</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>0.000000000000000888661812099</td>
<td>102</td>
<td>16606473870114816</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>51</td>
<td>0.000000000000000444330906049</td>
<td>104</td>
<td>33212947740229632</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>52</td>
<td>0.000000000000000222165453024</td>
<td>106</td>
<td>66425895480459264</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>53</td>
<td>0.000000000000000111082726512</td>
<td>108</td>
<td>132851790960918528</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Time to run for the B&P with flag algorithm (left) and number of processed boxes (right):

<table>
<thead>
<tr>
<th>k</th>
<th>nd</th>
<th>mbs</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.000222</td>
<td>0.61</td>
<td>16</td>
<td>15</td>
<td>14</td>
<td>13</td>
<td>12</td>
<td>11</td>
<td>10</td>
<td>9</td>
<td>8</td>
<td>7</td>
</tr>
<tr>
<td>1</td>
<td>0.038</td>
<td>0.074</td>
<td>0.263</td>
<td>0.821</td>
<td>2.410</td>
<td>6.85</td>
<td>18.18</td>
<td>48.834</td>
<td>130.159</td>
<td>362.059</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.038</td>
<td>0.034</td>
<td>0.488</td>
<td>5.61</td>
<td>15.845</td>
<td>3480.962</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.025</td>
<td>0.061</td>
<td>2.484</td>
<td>103.397</td>
<td>11329.79</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.062</td>
<td>0.058</td>
<td>4.99</td>
<td>1768.516</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0.031</td>
<td>0.075</td>
<td>23.23</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0.025</td>
<td>0.094</td>
<td>96.176</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0.007</td>
<td>0.112</td>
<td>481.307</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0.006</td>
<td>0.129</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>0.005</td>
<td>0.153</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0.004</td>
<td>0.174</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>0.004</td>
<td>0.180</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>0.00344</td>
<td>0.22</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>0.002122</td>
<td>0.232</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>0.016-0.258</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>0.05-0.286</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>1.530-0.324</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>7.636-0.368</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>1.891-0.39</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>3.816-0.432</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>9.451-0.458</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>4.776-0.505</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>2.388-0.557</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>1.196-0.596</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>4.566-0.662</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>2.986-0.687</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>1.498-0.722</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>7.456-0.776</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>1.376-0.832</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>1.866-0.888</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>9.316-0.948</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>4.666-1.006</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>2.336-1.067</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>1.166-1.159</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>5.826-1.211</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>2.796-1.273</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>1.466-1.334</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>7.286-1.410</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>3.846-1.477</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>1.835-1.557</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>9.066-1.635</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>4.556-1.717</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>2.276-1.822</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>43</td>
<td>1.146-1.912</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>44</td>
<td>5.688-1.972</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>45</td>
<td>2.846-2.076</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>46</td>
<td>1.426-2.142</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>47</td>
<td>7.166-2.248</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>48</td>
<td>3.556-2.352</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>49</td>
<td>1.786-2.444</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>8.886-2.572</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>51</td>
<td>4.446-2.686</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>52</td>
<td>2.226-2.798</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>k</th>
<th>nd</th>
<th>mbs</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.000222</td>
<td>0.61</td>
<td>16</td>
<td>15</td>
<td>14</td>
<td>13</td>
<td>12</td>
<td>11</td>
<td>10</td>
<td>9</td>
<td>8</td>
<td>7</td>
</tr>
<tr>
<td>1</td>
<td>0.038</td>
<td>0.074</td>
<td>0.263</td>
<td>0.821</td>
<td>2.410</td>
<td>6.85</td>
<td>18.18</td>
<td>48.834</td>
<td>130.159</td>
<td>362.059</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.062</td>
<td>0.058</td>
<td>4.99</td>
<td>1768.516</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.031</td>
<td>0.075</td>
<td>23.23</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.025</td>
<td>0.094</td>
<td>96.176</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0.007</td>
<td>0.112</td>
<td>481.307</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0.006</td>
<td>0.129</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0.005</td>
<td>0.153</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0.004</td>
<td>0.174</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>0.004</td>
<td>0.180</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0.00344</td>
<td>0.22</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>0.002122</td>
<td>0.232</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>0.016-0.258</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>0.05-0.286</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>1.530-0.324</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>7.636-0.368</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>1.891-0.39</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>3.816-0.432</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>9.451-0.458</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>4.776-0.505</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>2.388-0.557</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>1.196-0.596</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>4.566-0.662</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>2.986-0.687</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>1.498-0.722</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>7.456-0.776</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>1.376-0.832</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>1.866-0.888</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>9.316-0.948</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>4.666-1.006</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>2.336-1.067</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>1.166-1.159</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>5.826-1.211</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>2.796-1.273</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>1.466-1.334</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>7.286-1.410</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>3.846-1.477</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>1.835-1.557</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>9.066-1.635</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>4.556-1.717</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>2.276-1.822</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>1.146-1.912</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>5.688-1.972</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>43</td>
<td>2.846-2.076</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>44</td>
<td>1.426-2.142</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>45</td>
<td>7.166-2.248</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>46</td>
<td>3.556-2.352</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>47</td>
<td>1.786-2.444</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>48</td>
<td>8.886-2.572</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>49</td>
<td>4.446-2.686</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>2.226-2.798</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Time to run for the improved B&P algorithm (left) and number of processed boxes (right):

<table>
<thead>
<tr>
<th>k</th>
<th>time for B&amp;P (sec)</th>
<th>boxes processed</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.045</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0.078</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>0.097</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>0.126</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>0.153</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>0.181</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>0.209</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>0.235</td>
<td>8</td>
</tr>
<tr>
<td>9</td>
<td>0.262</td>
<td>9</td>
</tr>
<tr>
<td>10</td>
<td>0.287</td>
<td>10</td>
</tr>
<tr>
<td>11</td>
<td>0.312</td>
<td>11</td>
</tr>
<tr>
<td>12</td>
<td>0.337</td>
<td>12</td>
</tr>
<tr>
<td>13</td>
<td>0.362</td>
<td>13</td>
</tr>
<tr>
<td>14</td>
<td>0.387</td>
<td>14</td>
</tr>
<tr>
<td>15</td>
<td>0.413</td>
<td>15</td>
</tr>
<tr>
<td>16</td>
<td>0.438</td>
<td>16</td>
</tr>
<tr>
<td>17</td>
<td>0.464</td>
<td>17</td>
</tr>
<tr>
<td>18</td>
<td>0.488</td>
<td>18</td>
</tr>
<tr>
<td>19</td>
<td>0.513</td>
<td>19</td>
</tr>
<tr>
<td>20</td>
<td>0.538</td>
<td>20</td>
</tr>
<tr>
<td>21</td>
<td>0.563</td>
<td>21</td>
</tr>
<tr>
<td>22</td>
<td>0.588</td>
<td>22</td>
</tr>
<tr>
<td>23</td>
<td>0.613</td>
<td>23</td>
</tr>
<tr>
<td>24</td>
<td>0.638</td>
<td>24</td>
</tr>
<tr>
<td>25</td>
<td>0.663</td>
<td>25</td>
</tr>
<tr>
<td>26</td>
<td>0.688</td>
<td>26</td>
</tr>
<tr>
<td>27</td>
<td>0.713</td>
<td>27</td>
</tr>
<tr>
<td>28</td>
<td>0.738</td>
<td>28</td>
</tr>
<tr>
<td>29</td>
<td>0.763</td>
<td>29</td>
</tr>
<tr>
<td>30</td>
<td>0.788</td>
<td>30</td>
</tr>
<tr>
<td>31</td>
<td>0.813</td>
<td>31</td>
</tr>
<tr>
<td>32</td>
<td>0.838</td>
<td>32</td>
</tr>
<tr>
<td>33</td>
<td>0.863</td>
<td>33</td>
</tr>
<tr>
<td>34</td>
<td>0.888</td>
<td>34</td>
</tr>
<tr>
<td>35</td>
<td>0.913</td>
<td>35</td>
</tr>
<tr>
<td>36</td>
<td>0.938</td>
<td>36</td>
</tr>
<tr>
<td>37</td>
<td>0.963</td>
<td>37</td>
</tr>
<tr>
<td>38</td>
<td>0.988</td>
<td>38</td>
</tr>
<tr>
<td>39</td>
<td>1.013</td>
<td>39</td>
</tr>
<tr>
<td>40</td>
<td>1.038</td>
<td>40</td>
</tr>
<tr>
<td>41</td>
<td>1.063</td>
<td>41</td>
</tr>
<tr>
<td>42</td>
<td>1.088</td>
<td>42</td>
</tr>
<tr>
<td>43</td>
<td>1.113</td>
<td>43</td>
</tr>
<tr>
<td>44</td>
<td>1.138</td>
<td>44</td>
</tr>
<tr>
<td>45</td>
<td>1.163</td>
<td>45</td>
</tr>
<tr>
<td>46</td>
<td>1.188</td>
<td>46</td>
</tr>
<tr>
<td>47</td>
<td>1.213</td>
<td>47</td>
</tr>
<tr>
<td>48</td>
<td>1.238</td>
<td>48</td>
</tr>
<tr>
<td>49</td>
<td>1.263</td>
<td>49</td>
</tr>
<tr>
<td>50</td>
<td>1.288</td>
<td>50</td>
</tr>
<tr>
<td>51</td>
<td>1.313</td>
<td>51</td>
</tr>
<tr>
<td>52</td>
<td>1.338</td>
<td>52</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>k</th>
<th>time for B&amp;P (sec)</th>
<th>boxes processed</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.059</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0.083</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>0.107</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>0.131</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>0.155</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>0.179</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>0.203</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>0.227</td>
<td>8</td>
</tr>
<tr>
<td>9</td>
<td>0.251</td>
<td>9</td>
</tr>
<tr>
<td>10</td>
<td>0.275</td>
<td>10</td>
</tr>
<tr>
<td>11</td>
<td>0.300</td>
<td>11</td>
</tr>
<tr>
<td>12</td>
<td>0.324</td>
<td>12</td>
</tr>
<tr>
<td>13</td>
<td>0.349</td>
<td>13</td>
</tr>
<tr>
<td>14</td>
<td>0.373</td>
<td>14</td>
</tr>
<tr>
<td>15</td>
<td>0.398</td>
<td>15</td>
</tr>
<tr>
<td>16</td>
<td>0.422</td>
<td>16</td>
</tr>
<tr>
<td>17</td>
<td>0.447</td>
<td>17</td>
</tr>
<tr>
<td>18</td>
<td>0.471</td>
<td>18</td>
</tr>
<tr>
<td>19</td>
<td>0.496</td>
<td>19</td>
</tr>
<tr>
<td>20</td>
<td>0.520</td>
<td>20</td>
</tr>
<tr>
<td>21</td>
<td>0.545</td>
<td>21</td>
</tr>
<tr>
<td>22</td>
<td>0.570</td>
<td>22</td>
</tr>
<tr>
<td>23</td>
<td>0.595</td>
<td>23</td>
</tr>
<tr>
<td>24</td>
<td>0.620</td>
<td>24</td>
</tr>
<tr>
<td>25</td>
<td>0.644</td>
<td>25</td>
</tr>
<tr>
<td>26</td>
<td>0.669</td>
<td>26</td>
</tr>
<tr>
<td>27</td>
<td>0.694</td>
<td>27</td>
</tr>
<tr>
<td>28</td>
<td>0.719</td>
<td>28</td>
</tr>
<tr>
<td>29</td>
<td>0.744</td>
<td>29</td>
</tr>
<tr>
<td>30</td>
<td>0.769</td>
<td>30</td>
</tr>
<tr>
<td>31</td>
<td>0.794</td>
<td>31</td>
</tr>
<tr>
<td>32</td>
<td>0.819</td>
<td>32</td>
</tr>
<tr>
<td>33</td>
<td>0.844</td>
<td>33</td>
</tr>
<tr>
<td>34</td>
<td>0.869</td>
<td>34</td>
</tr>
<tr>
<td>35</td>
<td>0.894</td>
<td>35</td>
</tr>
<tr>
<td>36</td>
<td>0.919</td>
<td>36</td>
</tr>
<tr>
<td>37</td>
<td>0.944</td>
<td>37</td>
</tr>
<tr>
<td>38</td>
<td>0.969</td>
<td>38</td>
</tr>
<tr>
<td>39</td>
<td>0.994</td>
<td>39</td>
</tr>
<tr>
<td>40</td>
<td>1.019</td>
<td>40</td>
</tr>
<tr>
<td>41</td>
<td>1.044</td>
<td>41</td>
</tr>
<tr>
<td>42</td>
<td>1.069</td>
<td>42</td>
</tr>
<tr>
<td>43</td>
<td>1.094</td>
<td>43</td>
</tr>
<tr>
<td>44</td>
<td>1.119</td>
<td>44</td>
</tr>
<tr>
<td>45</td>
<td>1.144</td>
<td>45</td>
</tr>
<tr>
<td>46</td>
<td>1.169</td>
<td>46</td>
</tr>
<tr>
<td>47</td>
<td>1.194</td>
<td>47</td>
</tr>
<tr>
<td>48</td>
<td>1.219</td>
<td>48</td>
</tr>
<tr>
<td>49</td>
<td>1.244</td>
<td>49</td>
</tr>
<tr>
<td>50</td>
<td>1.269</td>
<td>50</td>
</tr>
<tr>
<td>51</td>
<td>1.294</td>
<td>51</td>
</tr>
<tr>
<td>52</td>
<td>1.319</td>
<td>52</td>
</tr>
</tbody>
</table>
Time to run (seconds) as a function of the minimum box size and number of dimensions for the B&C algorithm:

<table>
<thead>
<tr>
<th>A</th>
<th>Number of dimensions</th>
<th>Minimum Box size</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0.053</td>
</tr>
<tr>
<td>1</td>
<td>0.5</td>
<td>0.076</td>
</tr>
<tr>
<td>2</td>
<td>0.25</td>
<td>0.09</td>
</tr>
<tr>
<td>3</td>
<td>0.125</td>
<td>0.099</td>
</tr>
<tr>
<td>4</td>
<td>0.0625</td>
<td>0.132</td>
</tr>
<tr>
<td>5</td>
<td>0.03125</td>
<td>0.16</td>
</tr>
<tr>
<td>6</td>
<td>0.015625</td>
<td>0.192</td>
</tr>
<tr>
<td>7</td>
<td>0.007813</td>
<td>0.21</td>
</tr>
<tr>
<td>8</td>
<td>0.003906</td>
<td>0.238</td>
</tr>
<tr>
<td>9</td>
<td>0.001953</td>
<td>0.247</td>
</tr>
<tr>
<td>10</td>
<td>0.000977</td>
<td>0.266</td>
</tr>
</tbody>
</table>

| 11| 0.000488             | 0.289            |
|   | 0.000244             | 0.326            |
|   | 0.000122             | 0.342            |
|   | 8.3E-05              | 0.373            |
|   | 3.05E-05             | 0.391            |
|   | 1.53E-05             | 0.429            |
|   | 7.63E-06             | 0.452            |
|   | 3.81E-06             | 0.464            |
|   | 1.91E-06             | 0.5              |
|   | 9.54E-07             | 0.536            |
|   | 4.77E-07             | 0.555            |
|   | 2.38E-07             | 0.588            |
|   | 1.19E-07             | 0.603            |
|   | 5.96E-08             | 0.798            |
|   | 2.98E-08             | 0.851            |
|   | 1.49E-08             | 0.885            |
|   | 7.45E-09             | 0.902            |
|   | 3.73E-09             | 0.809            |
|   | 1.86E-09             | 0.798            |
|   | 9.35E-10             | 0.808            |
|   | 4.66E-10             | 0.832            |
|   | 2.33E-10             | 0.849            |
|   | 1.16E-10             | 0.934            |
|   | 5.82E-11             | 0.935            |
|   | 2.91E-11             | 0.936            |
|   | 1.46E-11             | 0.96             |
|   | 7.38E-12             | 1                |
|   | 3.64E-12             | 1.037            |
|   | 1.82E-12             | 1.122            |
|   | 9.09E-13             | 1.095            |
|   | 4.55E-13             | 1.194            |
|   | 2.27E-13             | 1.194            |
|   | 1.14E-13             | 1.202            |
|   | 5.64E-14             | 1.232            |
|   | 2.84E-14             | 1.246            |
|   | 1.42E-14             | 1.277            |
|   | 7.11E-15             | 1.369            |
|   | 3.55E-15             | 1.354            |
|   | 1.78E-15             | 1.417            |
|   | 8.84E-16             | 1.555            |
|   | 4.44E-16             | 1.581            |
|   | 2.22E-16             | 1.841            |
Number of boxes on the result as a function of the minimum box size and number of dimensions (left) and processed boxes during the algorithm as a function of the minimum box size and number of dimension (right) for the B&C algorithm:

<table>
<thead>
<tr>
<th>A</th>
<th>nbs</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>8</td>
<td>16</td>
<td>32</td>
<td>64</td>
<td>128</td>
<td>256</td>
<td>512</td>
<td>1024</td>
<td>2048</td>
</tr>
<tr>
<td>1</td>
<td>0.25</td>
<td>6</td>
<td>4</td>
<td>16</td>
<td>64</td>
<td>256</td>
<td>1024</td>
<td>4096</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.125</td>
<td>8</td>
<td>116</td>
<td>172</td>
<td>27600</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.0625</td>
<td>10</td>
<td>256</td>
<td>7880</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0.03125</td>
<td>12</td>
<td>512</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0.015625</td>
<td>14</td>
<td>1060</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0.007813</td>
<td>16</td>
<td>2112</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0.003906</td>
<td>18</td>
<td>4196</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>0.001953</td>
<td>20</td>
<td>8352</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0.000977</td>
<td>22</td>
<td>16628</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>0.000488</td>
<td>24</td>
<td>33256</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>0.000244</td>
<td>26</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>0.000122</td>
<td>28</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>6.1E-05</td>
<td>30</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>3.0E-05</td>
<td>32</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>1.5E-05</td>
<td>34</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>7.5E-06</td>
<td>36</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>3.8E-06</td>
<td>38</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>1.9E-06</td>
<td>40</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>9.5E-07</td>
<td>42</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>4.7E-07</td>
<td>44</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>2.3E-07</td>
<td>46</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>1.1E-07</td>
<td>48</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>5.9E-08</td>
<td>50</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>2.9E-08</td>
<td>52</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>1.4E-08</td>
<td>54</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>7.4E-09</td>
<td>56</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>3.7E-09</td>
<td>58</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>1.8E-09</td>
<td>60</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>9.3E-10</td>
<td>62</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>4.6E-10</td>
<td>64</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>2.3E-10</td>
<td>66</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>1.1E-10</td>
<td>68</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>5.8E-11</td>
<td>70</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>2.9E-11</td>
<td>72</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>1.4E-11</td>
<td>74</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>7.2E-12</td>
<td>76</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>3.6E-12</td>
<td>78</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>1.8E-12</td>
<td>80</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>9.0E-13</td>
<td>82</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>4.5E-13</td>
<td>84</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>2.2E-13</td>
<td>86</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>43</td>
<td>1.1E-13</td>
<td>88</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>44</td>
<td>5.6E-14</td>
<td>90</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>45</td>
<td>2.8E-14</td>
<td>92</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>46</td>
<td>1.4E-14</td>
<td>94</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>47</td>
<td>7.1E-15</td>
<td>96</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>48</td>
<td>3.6E-15</td>
<td>98</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>49</td>
<td>1.8E-15</td>
<td>100</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>9.0E-16</td>
<td>102</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>51</td>
<td>4.5E-16</td>
<td>104</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>52</td>
<td>2.2E-16</td>
<td>106</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A</th>
<th>nbs</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>8</td>
<td>16</td>
<td>32</td>
<td>64</td>
<td>128</td>
<td>256</td>
<td>512</td>
<td>1024</td>
<td>2048</td>
</tr>
<tr>
<td>1</td>
<td>0.25</td>
<td>6</td>
<td>16</td>
<td>64</td>
<td>256</td>
<td>1024</td>
<td>4096</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.125</td>
<td>8</td>
<td>116</td>
<td>172</td>
<td>27600</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.0625</td>
<td>10</td>
<td>256</td>
<td>7880</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0.03125</td>
<td>12</td>
<td>512</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0.015625</td>
<td>14</td>
<td>1060</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0.007813</td>
<td>16</td>
<td>2112</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0.003906</td>
<td>18</td>
<td>4196</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>0.001953</td>
<td>20</td>
<td>8352</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0.000977</td>
<td>22</td>
<td>16628</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>0.000488</td>
<td>24</td>
<td>33256</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>0.000244</td>
<td>26</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>0.000122</td>
<td>28</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>6.1E-05</td>
<td>30</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>3.0E-05</td>
<td>32</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>1.5E-05</td>
<td>34</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>7.5E-06</td>
<td>36</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>3.8E-06</td>
<td>38</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>1.9E-06</td>
<td>40</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>9.5E-07</td>
<td>42</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>4.7E-07</td>
<td>44</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>2.3E-07</td>
<td>46</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>1.1E-07</td>
<td>48</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>5.9E-08</td>
<td>50</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>2.9E-08</td>
<td>52</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>1.4E-08</td>
<td>54</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>7.4E-09</td>
<td>56</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>3.7E-09</td>
<td>58</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>1.8E-09</td>
<td>60</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>9.3E-10</td>
<td>62</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>4.6E-10</td>
<td>64</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>2.3E-10</td>
<td>66</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>1.1E-10</td>
<td>68</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>5.8E-11</td>
<td>70</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>2.9E-11</td>
<td>72</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>1.4E-11</td>
<td>74</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>7.2E-12</td>
<td>76</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>3.6E-12</td>
<td>78</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>1.8E-12</td>
<td>80</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>9.0E-13</td>
<td>82</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>4.5E-13</td>
<td>84</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>2.2E-13</td>
<td>86</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>43</td>
<td>1.1E-13</td>
<td>88</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>44</td>
<td>5.6E-14</td>
<td>90</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>45</td>
<td>2.8E-14</td>
<td>92</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>46</td>
<td>1.4E-14</td>
<td>94</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>47</td>
<td>7.1E-15</td>
<td>96</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>48</td>
<td>3.6E-15</td>
<td>98</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>49</td>
<td>1.8E-15</td>
<td>100</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>9.0E-16</td>
<td>102</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>51</td>
<td>4.5E-16</td>
<td>104</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>52</td>
<td>2.2E-16</td>
<td>106</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>