COMPUTER SOFTWARE ENGINEERING

JUNE 2020

DEVELOPMENT OF AGGREGATION METHODS OF PARTIALLY ORDERED SETS

CÉSAR GARCÍA CABEZA

TUTORS: IRENE DÍAZ RODRÍGUEZ & ELÍAS FERNÁNDEZ COMBARRO

TABLE OF CONTENTS

INTRODUCTION

BASIC CONCEPTS

ALGORITHMS

EXPERIMENTS

CONCLUSIONS

INTRODUCTION

REAL LIFE APPLICATION

PEPE

PEPA

PEPO

REAL LIFE APPLICATION

BEST

PEPA

PEPE

PEPO

WORST

MOTIVATION

N	COMBINATIONS
3	6
4	24
5	120
6	720
7	5040
8	40320
9	362880
10	3628800
11	39916800
12	479001600

NP-HARD[1]

[1] C. BACHMAIER ET AL. “ON THE HARDNESS OF MAXIMUM RANK AGGREGATION PROBLEMS” (2015)

BASIC CONCEPTS

PARTIALLY ORDERED SET OR *POSET*

SET P WITH A BINARY RELATION $\leq$ [2]

REFLEXIVITY

ANTISYMMETRY

TRANSITIVITY

2) CMU - DEPARTMENT OF MATHEMATICAL SCIENCES. *PARTIALLY ORDERED SETS* (2015)

PARTIALLY ORDERED SET OR *POSET*

SET P WITH A BINARY RELATION $\leq$[2]

REFLEXIVITY

ANTISYMMETRY

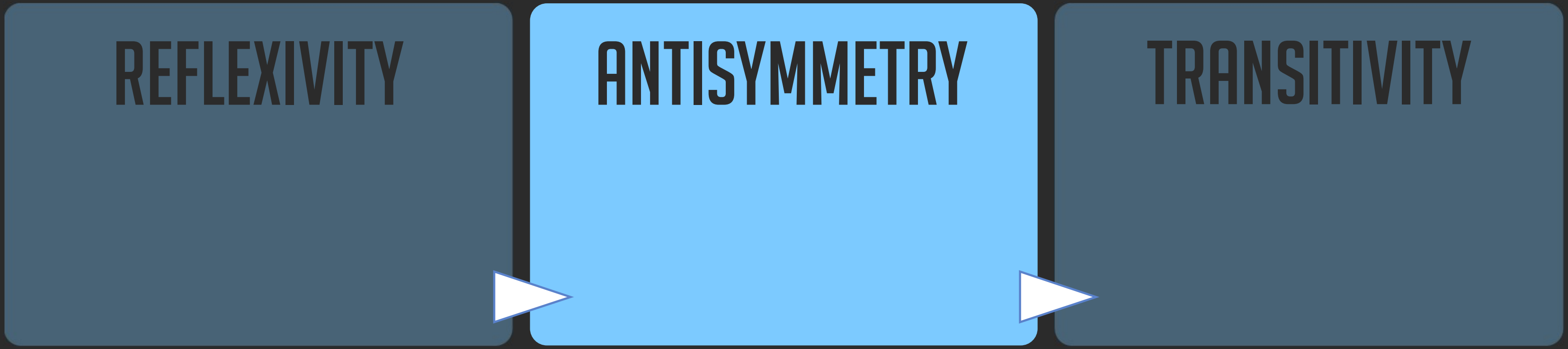
TRANSITIVITY

$$x \leq x$$

2) CMU - DEPARTMENT OF MATHEMATICAL SCIENCES. *PARTIALLY ORDERED SETS* (2015)

PARTIALLY ORDERED SET OR *POSET*

SET P WITH A BINARY RELATION $\leq$[2]



$$x \leq y, y \leq x \Rightarrow x = y$$

2) CMU - DEPARTMENT OF MATHEMATICAL SCIENCES. *PARTIALLY ORDERED SETS* (2015)

PARTIALLY ORDERED SET OR *POSET*

SET P WITH A BINARY RELATION $\leq$[2]

REFLEXIVITY

ANTISYMMETRY

TRANSITIVITY

$$x \leq y, y \leq z \Rightarrow x \leq z$$

2) CMU - DEPARTMENT OF MATHEMATICAL SCIENCES. *PARTIALLY ORDERED SETS* (2015)

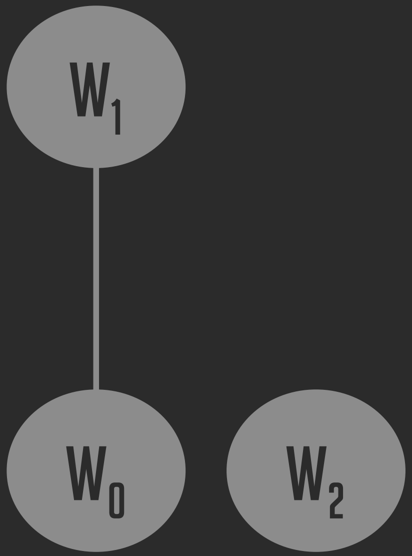
COMPARABLE OBJECTS

$x \leq y$

OR

$y \leq x$

POSET REPRESENTATION



$$\equiv$$

$$\begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

HASSE DIAGRAM[3]

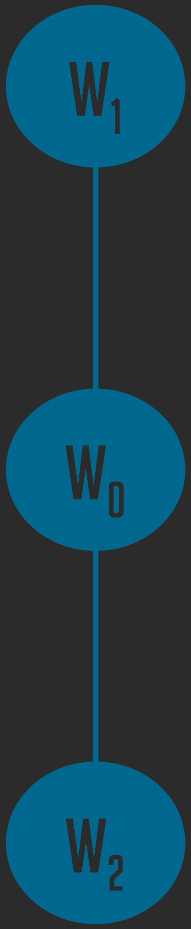
MATRIX

3) E.F. COMBARRO, I. DÍAZ AND P. MIRANDA. "ON RANDOM GENERATION OF FUZZY MEASURES" (2013)

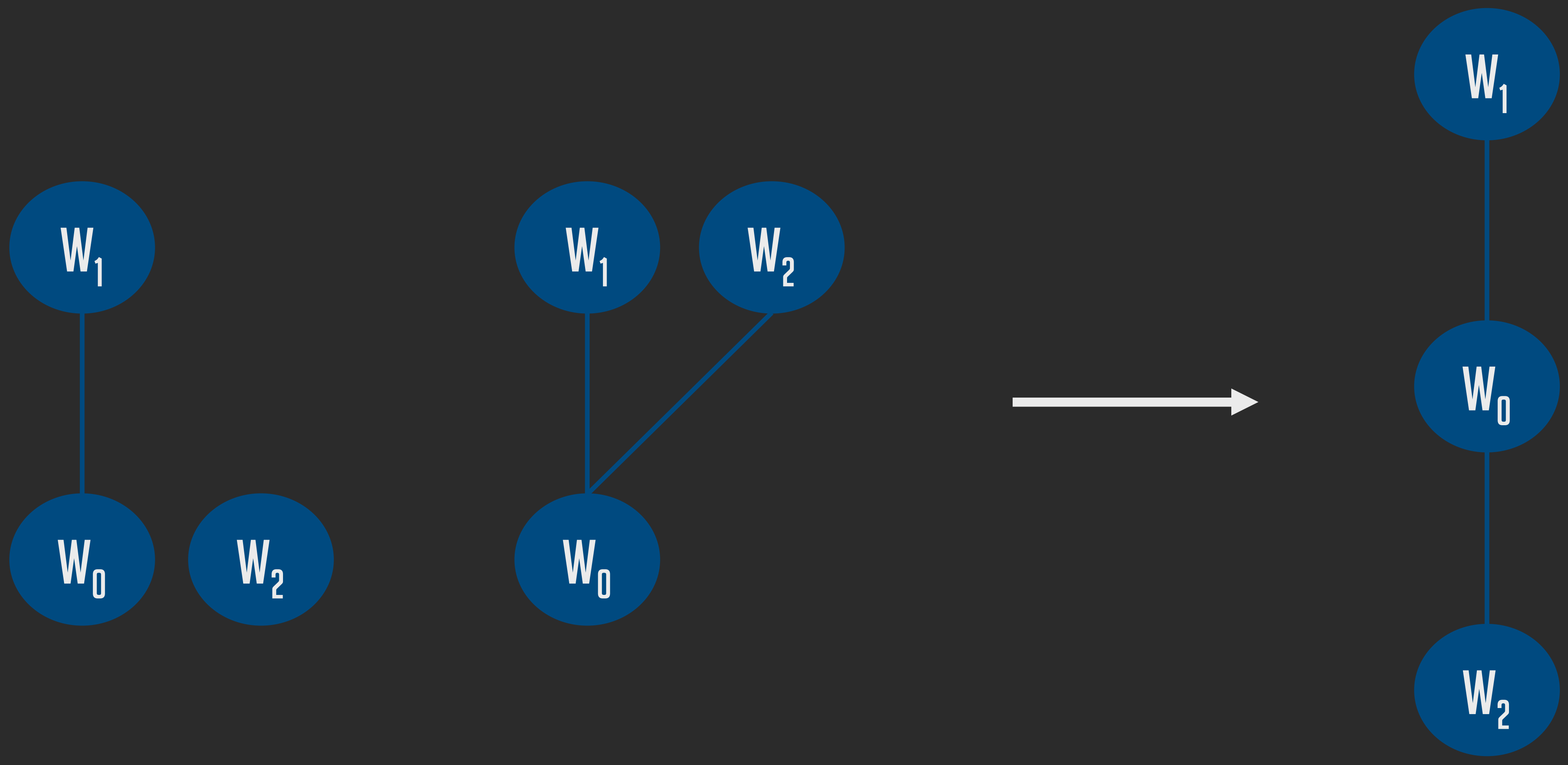
LINEAR EXTENSION

TOTAL ORDER RELATIONSHIP

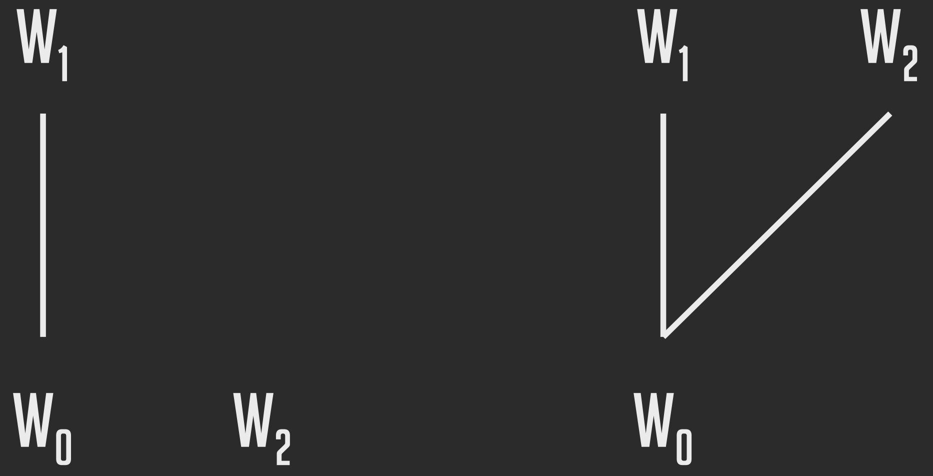
ALL ELEMENTS ARE COMPARABLE TO EACH OTHER



AGGREGATION OF *POSETS*



AGGREGATION MATRIX



AGGREGATION MATRIX

$$\begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

AGGREGATION MATRIX

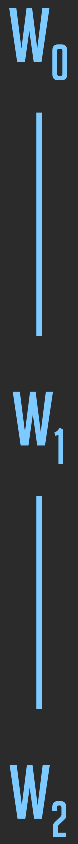
$$A = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} + \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 2 & 2 & 1 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{pmatrix}$$

COST OF AN AGGREGATION

NOT OPTIMAL

PARTIAL RESTRICTIONS VIOLATED BY THE LINEAR EXTENSION

$$\begin{pmatrix} 2 & 2 & 1 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{pmatrix}$$



w_0

w_1

w_2

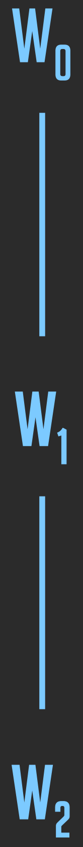
COST =

COST OF AN AGGREGATION

NOT OPTIMAL

PARTIAL RESTRICTIONS VIOLATED BY THE LINEAR EXTENSION

$$\begin{pmatrix} 2 & 2 & 1 \\ 0 & 2 & \mathbf{0} \\ 0 & 0 & 2 \end{pmatrix}$$



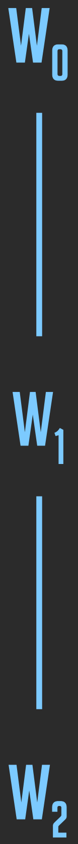
COST = 0

COST OF AN AGGREGATION

NOT OPTIMAL

PARTIAL RESTRICTIONS VIOLATED BY THE LINEAR EXTENSION

$$\begin{pmatrix} 2 & 2 & 1 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{pmatrix}$$



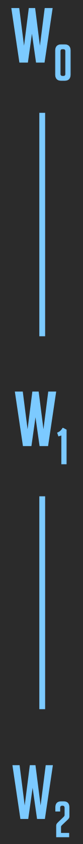
COST = 0 + 1

COST OF AN AGGREGATION

NOT OPTIMAL

PARTIAL RESTRICTIONS VIOLATED BY THE LINEAR EXTENSION

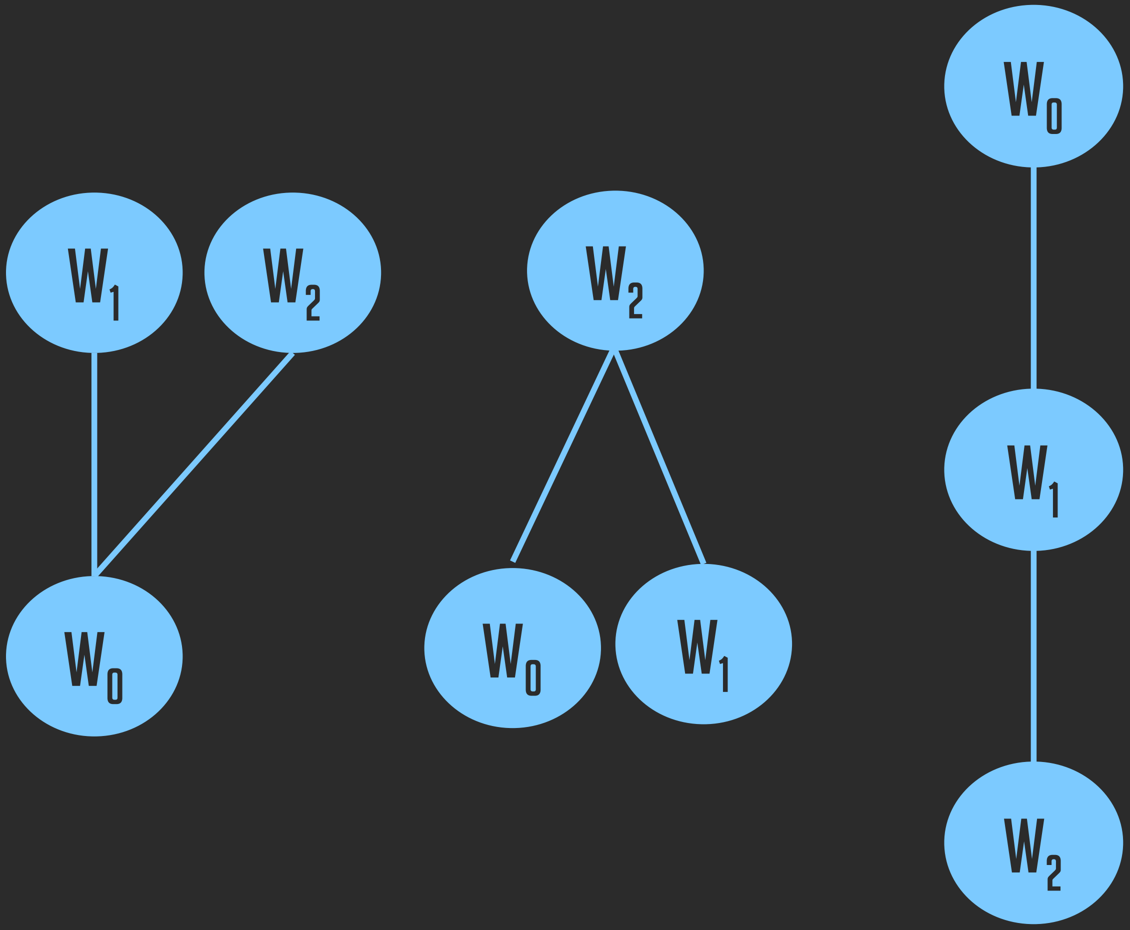
$$\begin{pmatrix} 2 & \mathbf{2} & 1 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{pmatrix}$$



COST = 0 + 1 + 2 = 3

ALGORITHMS

EXAMPLE



$$A = \begin{pmatrix} 3 & 1 & 2 \\ 1 & 3 & 1 \\ 1 & 1 & 3 \end{pmatrix}$$

MINCOST ST

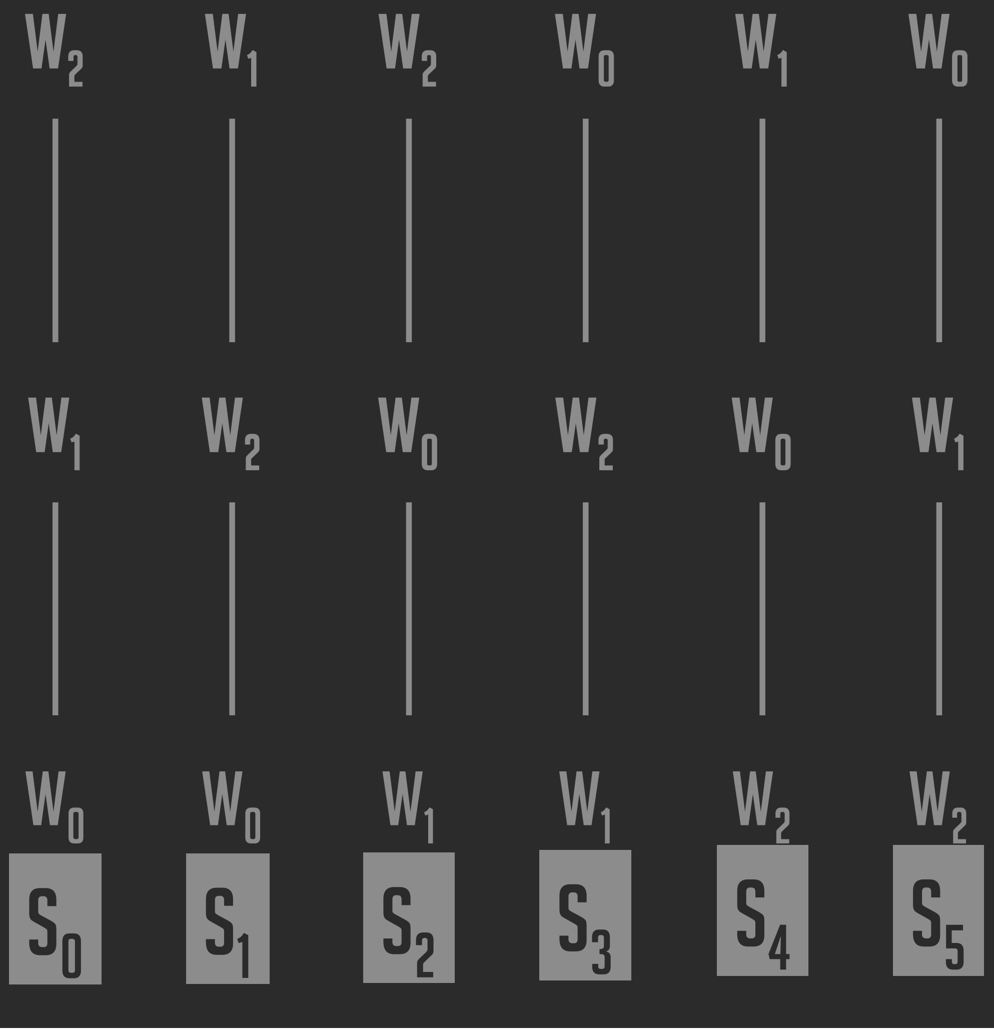
COMPUTING ALL THE POSIBLE LINEAR EXTENSIONS AND KEEPING THE BEST

SEQUENTIALLY CALCULATES THE COST

OPTIMAL ALGORITHM

HIGH EXECUTION TIMES

MINCOST ST - EXAMPLE



W_2	W_1	W_2	W_0	W_1	W_0
W_1	W_2	W_0	W_2	W_0	W_1
W_0	W_0	W_1	W_1	W_2	W_2
S_0	S_1	S_2	S_3	S_4	S_5

POSSIBLE SOLUTION	COST
S_0	3
S_1	3
S_2	4
S_3	3
S_4	4
S_5	4

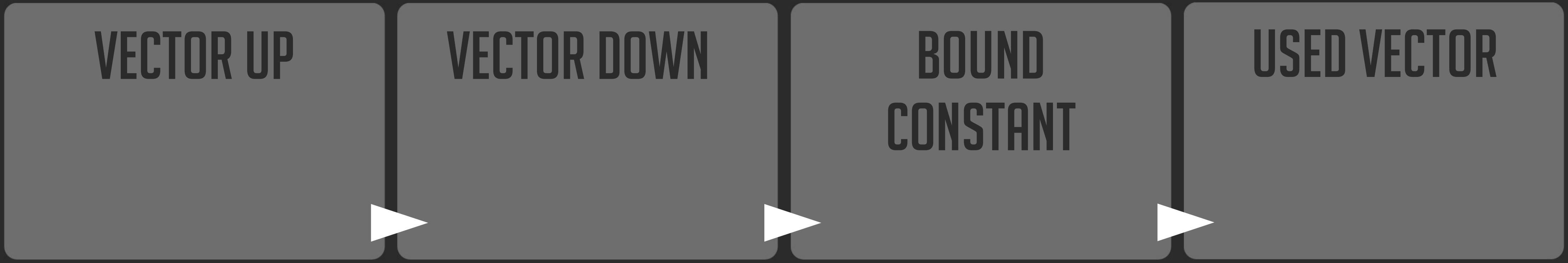
Minimals[4]

What is a *minimal* element?

An element $a \in P$ is a *minimal* element if there is no $b \in P$ such that $a > b$

[4] E.F. Combarro, J.H. de Saracho and I.D. Rodríguez. “Minimals Plus: an improved...” (2019)

MINIMALS - INITIALIZATION

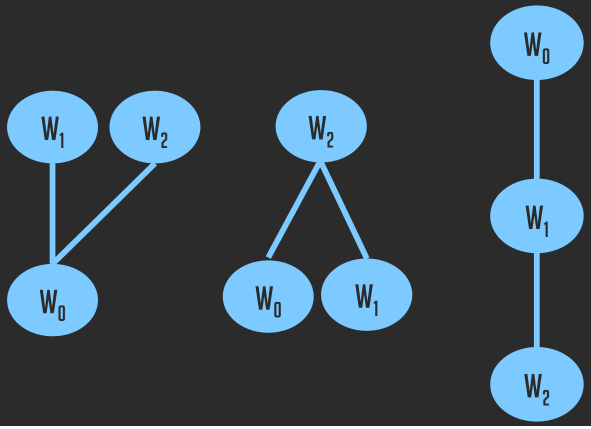


VECTOR UP

VECTOR DOWN

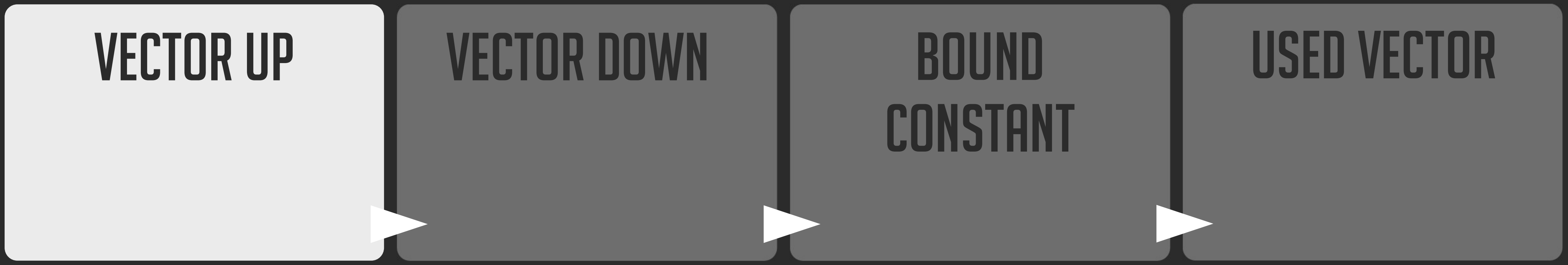
BOUND CONSTANT

USED VECTOR



$$A = \begin{pmatrix} 3 & 1 & 2 \\ 1 & 3 & 1 \\ 1 & 1 & 3 \end{pmatrix}$$

MINIMALS - INITIALIZATION



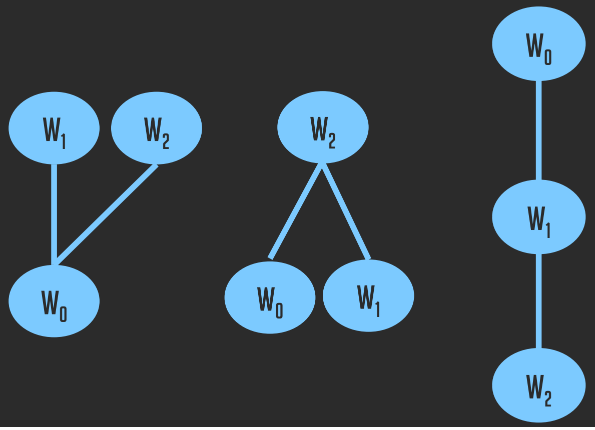
VECTOR UP

VECTOR DOWN

BOUND CONSTANT

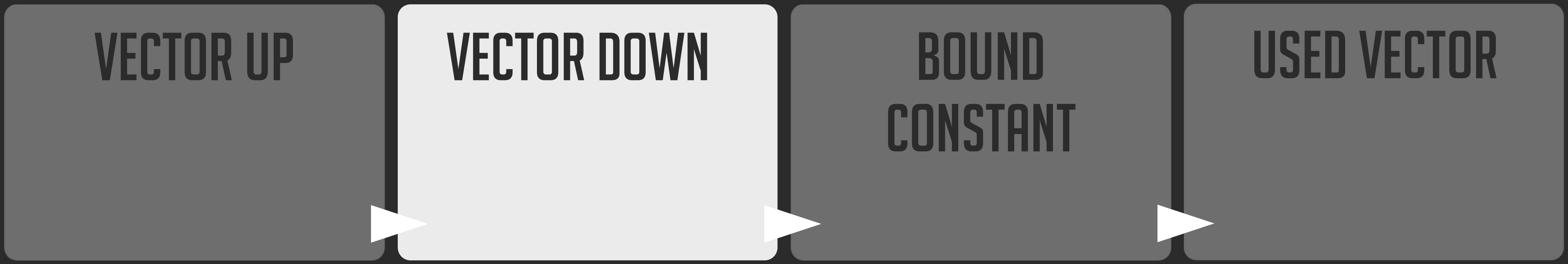
USED VECTOR

$$up[i] = \sum_{j}^{n} A[i,j] \quad up = [6,5,5]$$



$$A = \begin{pmatrix} 3 & 1 & 2 \\ 1 & 3 & 1 \\ 1 & 1 & 3 \end{pmatrix}$$

MINIMALS - INITIALIZATION



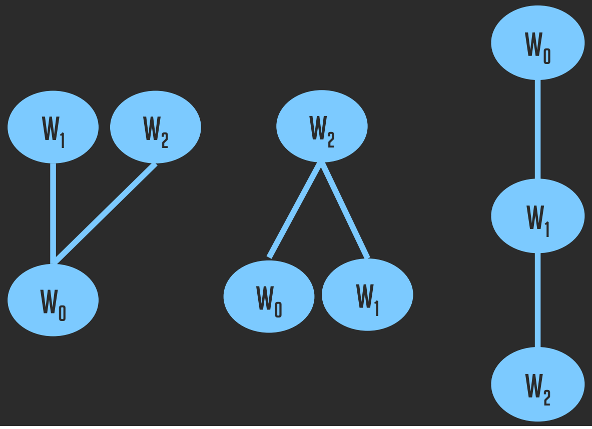
VECTOR UP

VECTOR DOWN

BOUND CONSTANT

USED VECTOR

$\text{down}[i] = \sum_j^n A[j,i]$ down= $[5,5,6]$



$$A = \begin{pmatrix} 3 & 1 & 2 \\ 1 & 3 & 1 \\ 1 & 1 & 3 \end{pmatrix}$$

MINIMALS - INITIALIZATION

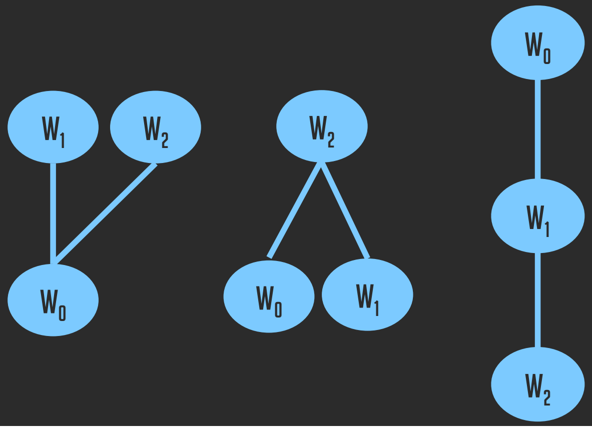
VECTOR UP

VECTOR DOWN

BOUND CONSTANT

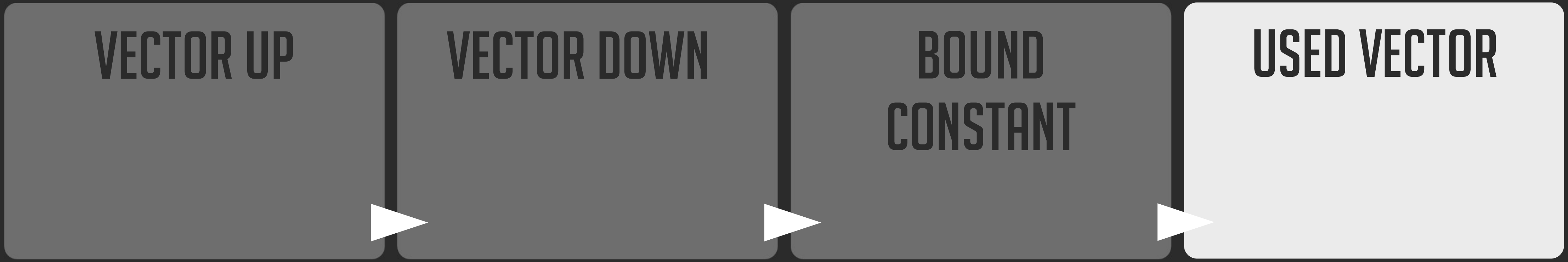
USED VECTOR

$$\text{bound} = \sum_{j}^{n} up[i] \qquad bound = 16$$



$$A = \begin{pmatrix} 3 & 1 & 2 \\ 1 & 3 & 1 \\ 1 & 1 & 3 \end{pmatrix}$$

MINIMALS - INITIALIZATION



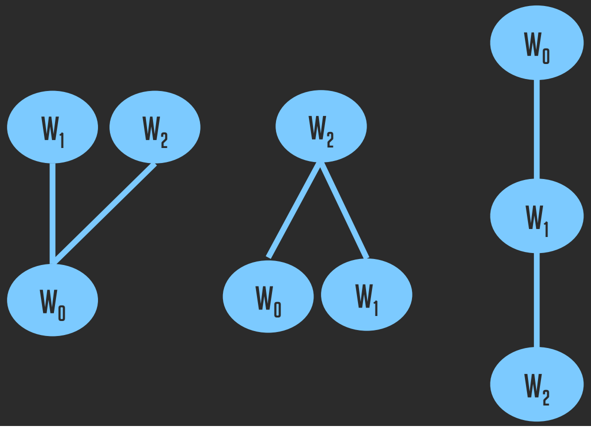
VECTOR UP

VECTOR DOWN

BOUND CONSTANT

USED VECTOR

$$used = [False, False, False]$$



$$A = \begin{pmatrix} 3 & 1 & 2 \\ 1 & 3 & 1 \\ 1 & 1 & 3 \end{pmatrix}$$

MINIMALS – SEARCH OF THE MINIMALS

1º) LOWEST NUMBER OF ELEMENTS BELOW

MIN = 5

$$P(i) = \frac{up[i]}{\sum_{j\ minimal} up[i]}$$

2º) MINIMALS

MINIMALS = $[W_0, W_1]$

3º) CHOOSE MINIMAL

PROBABILITIES = $[\frac{6}{11}, \frac{5}{11}]$

W_1

4º) UPDATE

$$A = \begin{pmatrix} 3 & 1 & 2 \\ 1 & 3 & 1 \\ 1 & 1 & 3 \end{pmatrix}$$

	W_0	W_1	W_2
UP	6	5	5
DOWN	5	5	6
USED	FALSE	FALSE	FALSE

BOUND = 16

MINIMALS – SEARCH OF THE MINIMALS

1º) LOWEST NUMBER OF ELEMENTS BELOW

MIN = 5

2º) *MINIMALS*

MINIMALS = [W_0, W_1]

3º) CHOOSE *MINIMAL*

PROBABILITIES = $[\frac{6}{11}, \frac{5}{11}]$

W_1

4º) UPDATE

$$A = \begin{pmatrix} 3 & 1 & 2 \\ 1 & 3 & 1 \\ 1 & 1 & 3 \end{pmatrix}$$

	W_0	W_1	W_2
UP	6-1 = 5	5-3 = 2	5-1 = 4
DOWN	5-1 = 4	5-3 = 2	6-1=5
USED	FALSE	TRUE	FALSE

BOUND = 16

Minimals – Search of the minimals

1º) Lowest number of elements below

Min = 4

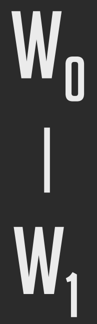
2º) *Minimals*

Minimals = [W_0]

3º) Choose *minimal*

Probabilities = [1]

4º) Update



$$A = \begin{pmatrix} 3 & 1 & 2 \\ 1 & 3 & 1 \\ 1 & 1 & 3 \end{pmatrix}$$

	W_0	W_1	W_2
Up	5	2	4
Down	4	2	5
Used	False	True	False

Bound = 16

MINIMALS – SEARCH OF THE MINIMALS

1º) LOWEST NUMBER OF ELEMENTS BELOW

MIN = 4

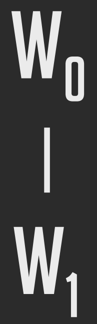
2º) *MINIMALS*

MINIMALS = [W_0]

3º) CHOOSE *MINIMAL*

PROBABILITIES = [1]

4º) UPDATE



$$A = \begin{pmatrix} 3 & 1 & 2 \\ 1 & 3 & 1 \\ 1 & 1 & 3 \end{pmatrix}$$

	W_0	W_1	W_2
UP	5-3 = 2	2-1 = 1	4-1 = 3
DOWN	4-3 = 1	2-1 = 1	5-2 = 3
USED	TRUE	TRUE	FALSE

BOUND = 16

MINIMALS – SEARCH OF THE MINIMALS

1º) LOWEST NUMBER OF ELEMENTS BELOW

MIN = 3

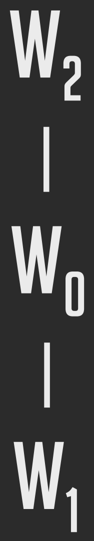
2º) *MINIMALS*

MINIMALS = [W_2]

3º) CHOOSE *MINIMAL*

PROBABILITIES = [1]

4º) UPDATE



$$A = \begin{pmatrix} 3 & 1 & 2 \\ 1 & 3 & 1 \\ 1 & 1 & 3 \end{pmatrix}$$

	W_0	W_1	W_2
UP	2	1	3
DOWN	1	1	3
USED	TRUE	TRUE	FALSE

BOUND = 16

MINIMALS – SEARCH OF THE MINIMALS

1º) LOWEST NUMBER OF ELEMENTS BELOW

MIN = 3

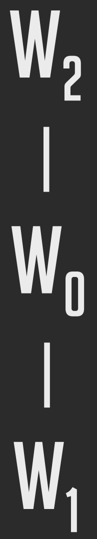
2º) *MINIMALS*

MINIMALS = [W_2]

3º) CHOOSE *MINIMAL*

PROBABILITIES = [1]

4º) UPDATE



W_2

W_0

W_1

$$A = \begin{pmatrix} 3 & 1 & 2 \\ 1 & 3 & 1 \\ 1 & 1 & 3 \end{pmatrix}$$

	W_0	W_1	W_2
UP	2-2=0	1-1=0	3-3=0
DOWN	1-1=0	1-1=0	3-3=0
USED	TRUE	TRUE	TRUE

BOUND = 16

MINIMALS RANDOM

RANDOMLY CHOSEN

~~VECTOR UP~~

MINIMALS RANDOM - EXAMPLE

MINIMALS

MINIMALS = [W_0, W_1]

PROBABILITIES = $[\frac{6}{11}, \frac{5}{11}]$

MINIMALS RANDOM

MINIMALS = [W_0, W_1]

PROBABILITIES = $[\frac{1}{2}, \frac{1}{2}]$

$$P(i) = P(i) = \frac{1}{k}\,\frac{{}^{[i]}}{\imath l\; up[i]}$$

MINCOST MT

BASED ON MINCOST ST

PARALLEL[5]

[5] E. OUELLET AND O. SAAD "FAST IMPLEMENTATIONS AND A NEW INDEXING..." (2018)

SORTING ALGORITHMS

- BUBBLE
- SELECTION
- INSERTION
- QUICKSORT
- MERGESORT

SORTING_METHOD(WHAT_TO_ORDER, COMPARATOR)

SORTING ALGORITHMS – COMPARISON A

ONLY TAKE INTO ACCOUNT THE TIMES AN *I* ELEMENT IS LOWER OR GREATER THAN ANOTHER *J*

$$P\,(I \leq J) = \begin{cases} \frac{lower}{lower+greater} & if\ lower + greater \neq 0 \\ 0.5 & otherwise \end{cases}$$

LOWER → A[I, J]

GREATER → A[J, I]

SORTING ALGORITHMS – COMPARISON B

THE TIMES AN *I* ELEMENT IS LOWER OR GREATER THAN ANOTHER *J*

THE TIMES OBJECTS I AND J ARE NOT COMPARABLE

$$P\,(I \leq J) = \left\{ \frac{lower \quad .5\ x\ notCompared}{total} \right.$$

LOWER → A[I, J]   GREATER → A[J, I]

TOTAL → A[I, I]

NOT COMPARED → TOTAL – (LOWER + GREATER)

SORTING ALGORITHMS – EXAMPLE

$$A = \begin{bmatrix} 3 & 1 & 2 \\ 1 & 3 & 1 \\ 1 & 1 & 3 \end{bmatrix}$$

COMPARISON A

$$\begin{bmatrix} 1 & \frac{A[0,1]}{A[0,1] + A[1,0]} & \frac{A[0,2]}{A[0,2] + A[2,0]} \\ \frac{A[1,0]}{A[1,0] + A[0,1]} & 1 & \frac{A[1,2]}{A[1,2] + A[2,1]} \\ \frac{A[2,0]}{A[2,0] + A[0,2]} & \frac{A[2,1]}{A[2,1] + A[1,2]} & 1 \end{bmatrix}$$

COMPARISON B

$$\begin{bmatrix} 1 & \frac{A[0,1] + 0.5\ x\ 1}{3} & \frac{A[0,2] + 0.5\ x\ 0}{3} \\ \frac{A[1,0] + 0.5\ x\ 1}{3} & 1 & \frac{A[1,2] + 0.5\ x\ 1}{3} \\ \frac{A[2,0] + 0.5\ x\ 0}{3} & \frac{A[2,1] + 0.5\ x\ 1}{3} & 1 \end{bmatrix}$$

SORTING ALGORITHMS – EXAMPLE

$$A = \begin{bmatrix} 3 & 1 & 2 \\ 1 & 3 & 1 \\ 1 & 1 & 3 \end{bmatrix}$$

COMPARISON A

$$\begin{bmatrix} 1 & \frac{1}{2} & \frac{2}{3} \\ \frac{1}{2} & 1 & \frac{1}{2} \\ \frac{1}{3} & \frac{1}{2} & 1 \end{bmatrix}$$

COMPARISON B

$$\begin{bmatrix} 1 & \frac{1}{2} & \frac{2}{3} \\ \frac{1}{2} & 1 & \frac{1}{2} \\ \frac{1}{3} & \frac{1}{2} & 1 \end{bmatrix}$$

SIMULATED ANNEALING[6]

OPTIMIZATION ALGORITHM → INITIAL SOLUTION

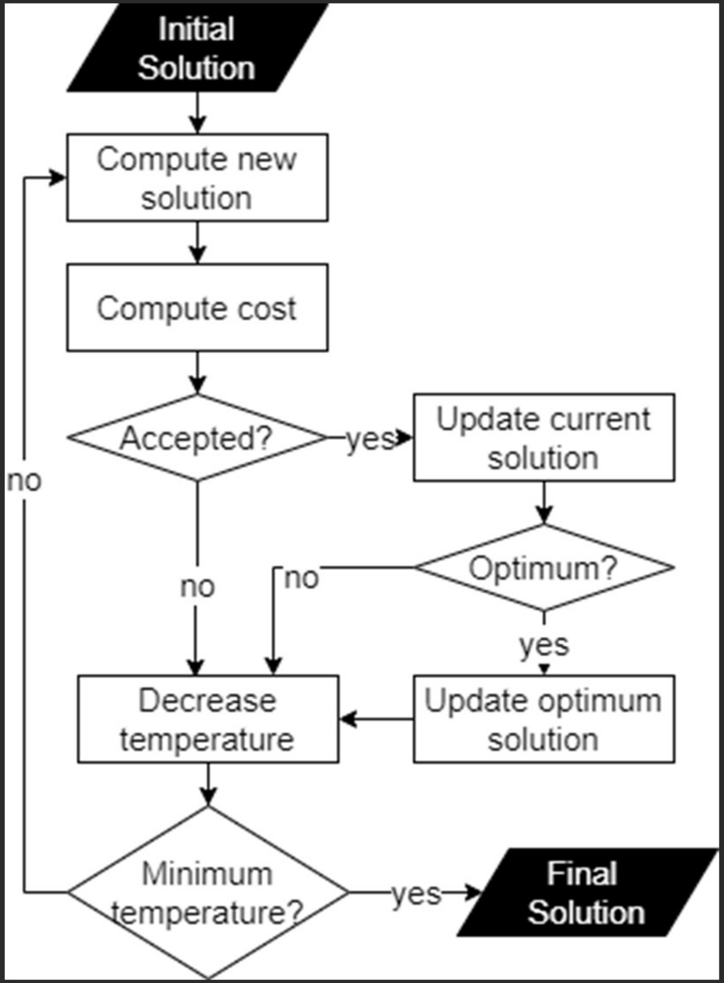
[6] D.T. PHAM AND D. KARABOGA "INTELLIGENT OPTIMISATION TECHNIQUES: GENETICS..." (2000)

SIMULATED ANNEALING - FLOWCHART

COMPUTE NEW SOLUTION

COMPUTE COST

ACCEPT AND UPDATE



UPDATE OPTIMUM

LOWER TEMPERATURE

SIMULATED ANNEALING – CALCULATE NEW SOLUTION

SWAP TWO POSITIONS

$[W_0, W_2, W_1]$

$[W_1, W_2, W_0]$

I = RANDOM(N) = 2

SIMULATED ANNEALING – CALCULATE COST SOLUTION

ALGORITHM OF THE COST

SIMULATED ANNEALING – ACCEPT AND UPDATE

NEWCOST <= CURRENTCOST

$$P(accepted) = 1$$

NEWCOST > CURRENTCOST

$$P^{7}(accepted) = e^{\frac{-\Delta cost}{T}}$$

7) J. Dréo et al. "Metaheuristics for Hard Optimization: Simulated Annealing, ..." (2006)

SIMULATED ANNEALING – UPDATE BEST

KEEP BEST SOLUTION

SIMULATED ANNEALING – LOWER TEMPERATURE

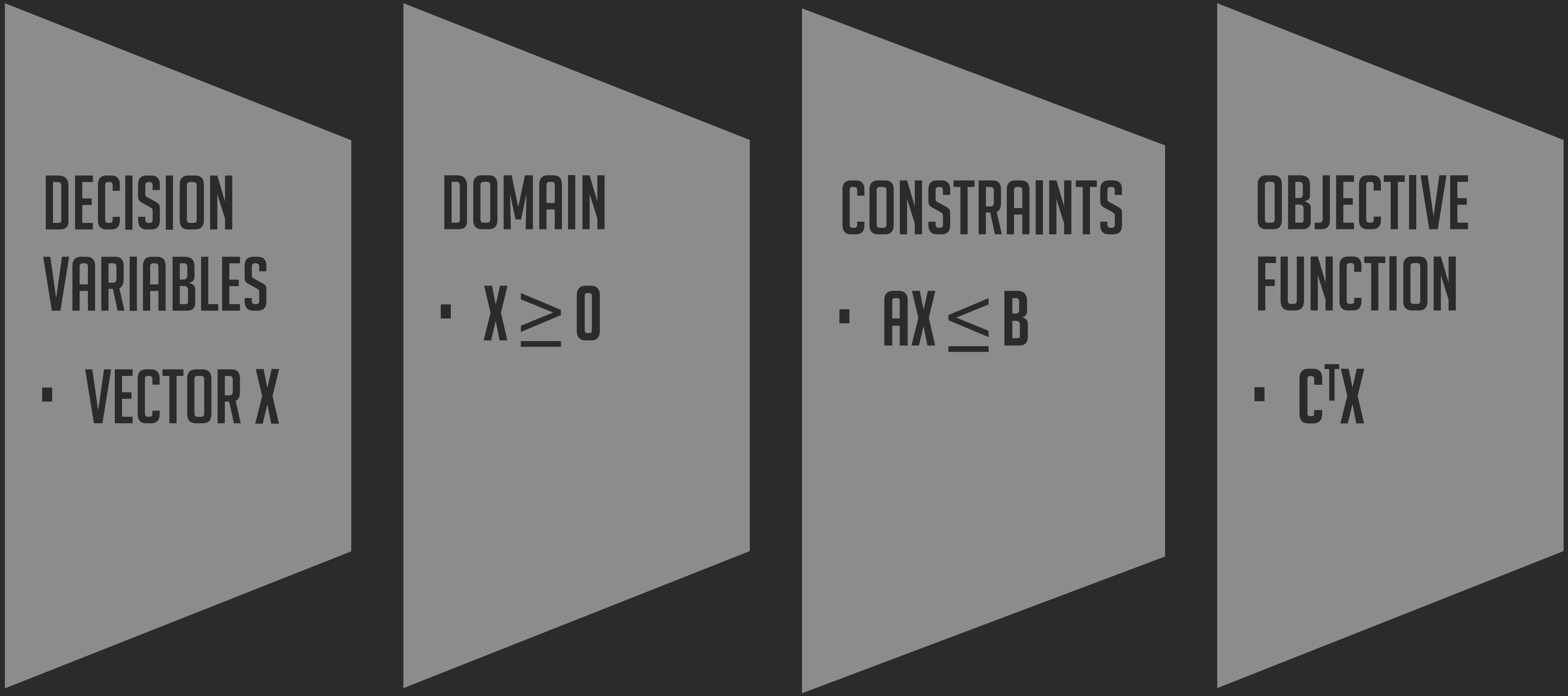
INITIAL TEMPERATURE

COOLING SYSTEM

LIMIT TEMPERATURE

$$T_{i+1} = \beta\, T_i$$

LINEAR PROGRAMMING – 4 CONCEPTS[6]



DECISION VARIABLES

- VECTOR X

DOMAIN

- $X \geq 0$

CONSTRAINTS

- $AX \leq B$

OBJECTIVE FUNCTION

- $C^{T}X$

[6] A. VIDHYA. "INTRODUCTORY GUIDE ON LINEAR PROGRAMMING" (2020)

LINEAR PROGRAMMING – STANDARD FORM[7]

$\max\{c^T \mid Ax \leq b \wedge x \geq 0\}$

$\min\{c^T \mid Ax \leq b \wedge x \geq 0\}$

[7] O.A. CAMARENA. "LINEAR PROGRAMMING" (2018)

LINEAR PROGRAMMING – EXAMPLE

VARIABLES

3 OBJECTS

V MATRIX N X N SIZE

$$V = \begin{bmatrix} V_{o,o} & V_{o,1} & V_{o,2} \\ V_{1,0} & V_{1,1} & V_{1,2} \\ V_{2,0} & V_{2,1} & V_{2,2} \end{bmatrix}$$

LINEAR PROGRAMMING – EXAMPLE

DOMAIN

BINARY

ZERO-ONE LINEAR PROGRAMMING

$$D = \{0, 1\}$$

LINEAR PROGRAMMING – EXAMPLE

CONSTRAINTS

CONSTRAINT	TYPE
V[0,0] = 1	DIAGONAL
V[1,1] = 1	DIAGONAL
V[2,2] = 1	DIAGONAL
V[0,1] + V[1,0] = 1	NO CYCLES
V[0,2] + V[2,0] = 1	NO CYCLES
V[2,1] + V[1,2] = 1	NO CYCLES
V[0,1] + V[1,2] - V[0,2] ≤ 1	TRANSITIVITY
V[0,2] + V[2,1] - V[0,1] ≤ 1	TRANSITIVITY
V[1,0] + V[0,2] - V[1,2] ≤ 1	TRANSITIVITY
V[1,2] + V[2,0] - V[1,0] ≤ 1	TRANSITIVITY
V[2,0] + V[0,1] - V[2,1] ≤ 1	TRANSITIVITY
V[2,1] + V[1,0] - V[2,0] ≤ 1	TRANSITIVITY

LINEAR PROGRAMMING – EXAMPLE

OBJECTIVE FUNCTION

COST OF THE AGGREGATION

MINIMISE

VARIABLE X PARTIAL COST

$$objectiveFunction(V, A) = \sum_{i,j}^{n} V[i,j]x\ A[j,i] = V[0,1]x\ A[1,0] + V[0,2]x\ A[2,0] + \ \ldots + V[2,1]x\ A[1,2]$$

$$\forall i \neq j$$

EXPERIMENTS

PARAMETERS

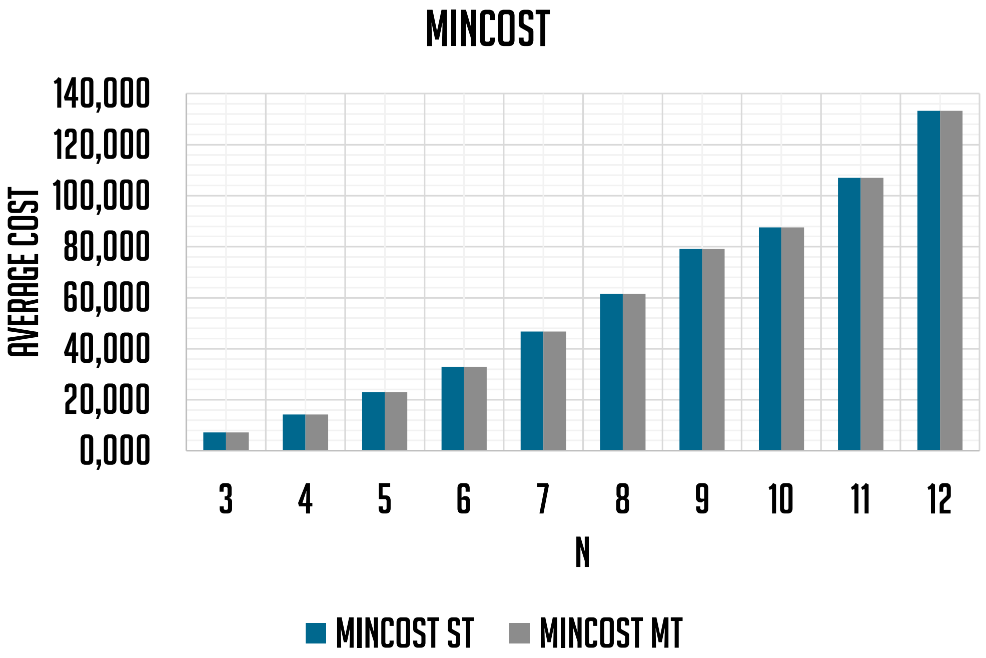
SIZE OF THE POSETS

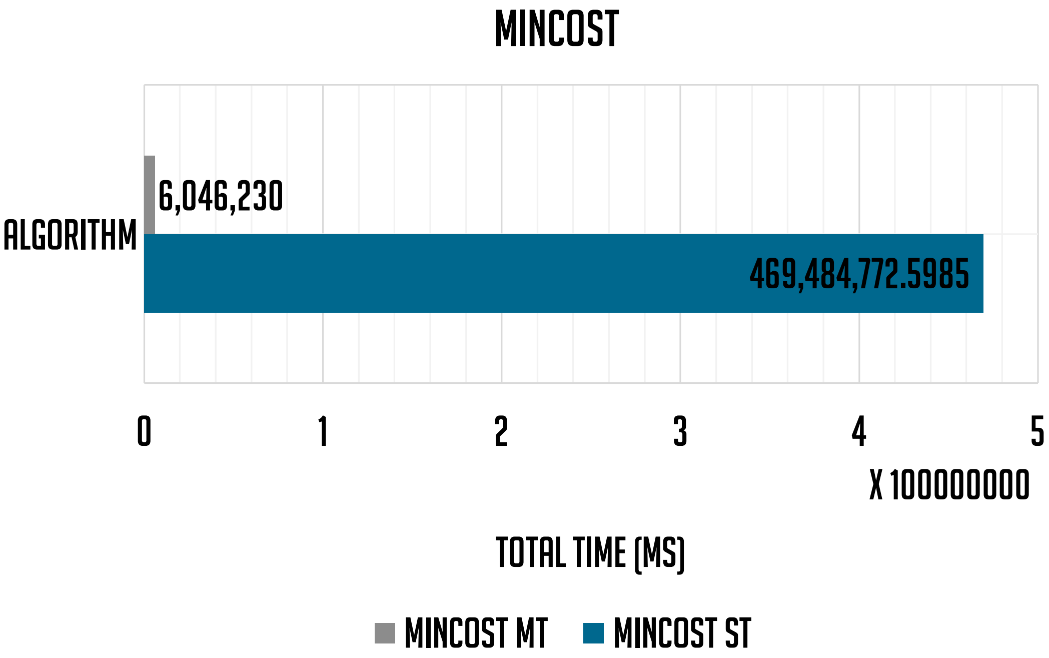
AMOUNT OF POSETS

RESULTS

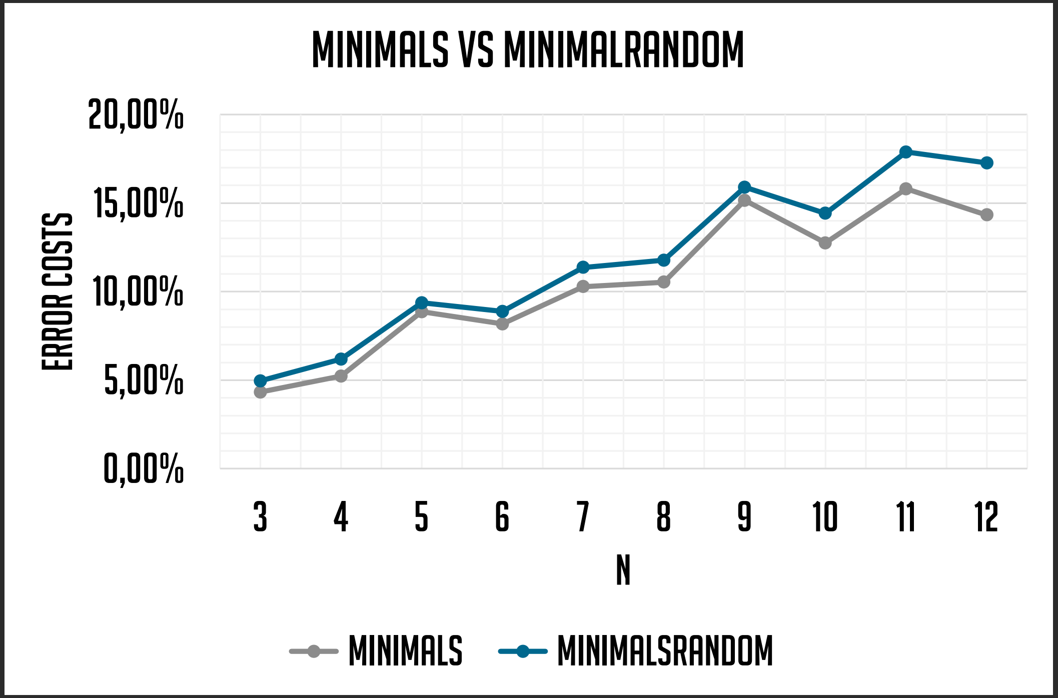
AVERAGE

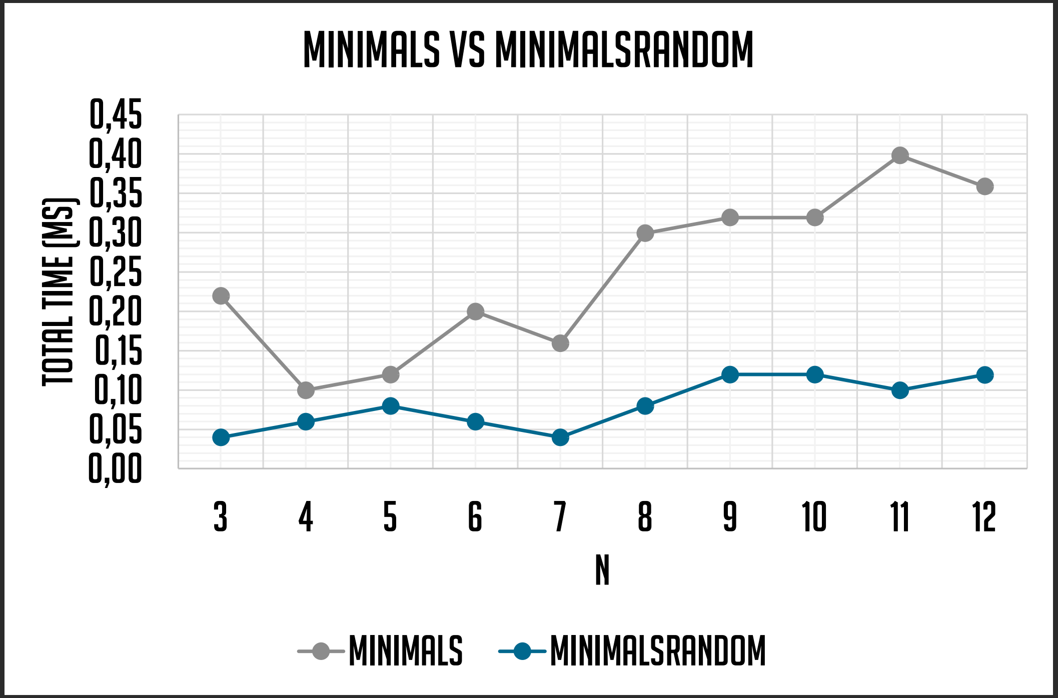
MINCOST ST VS MINCOST MT



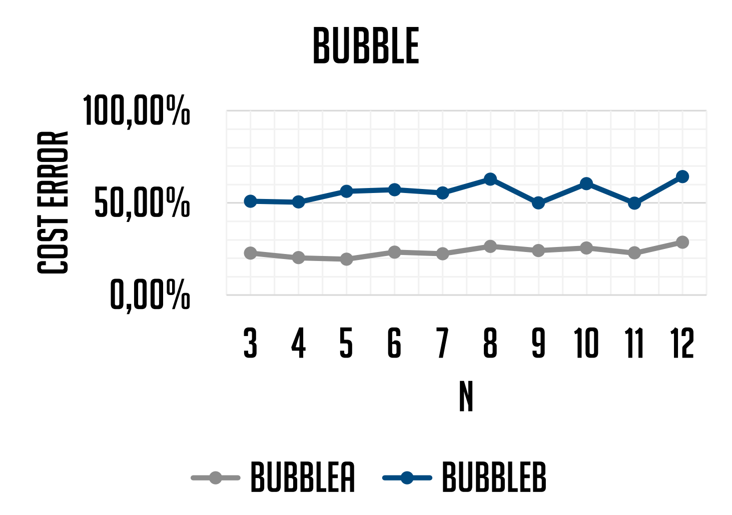


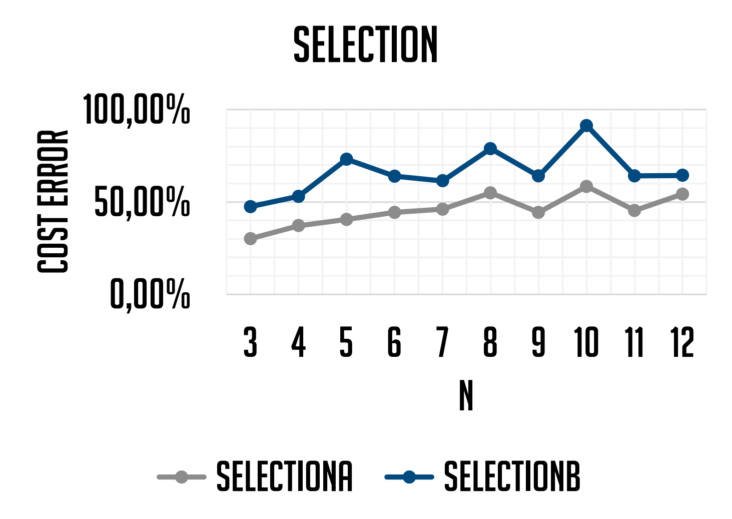
MINIMALS VS MINIMALS RANDOM

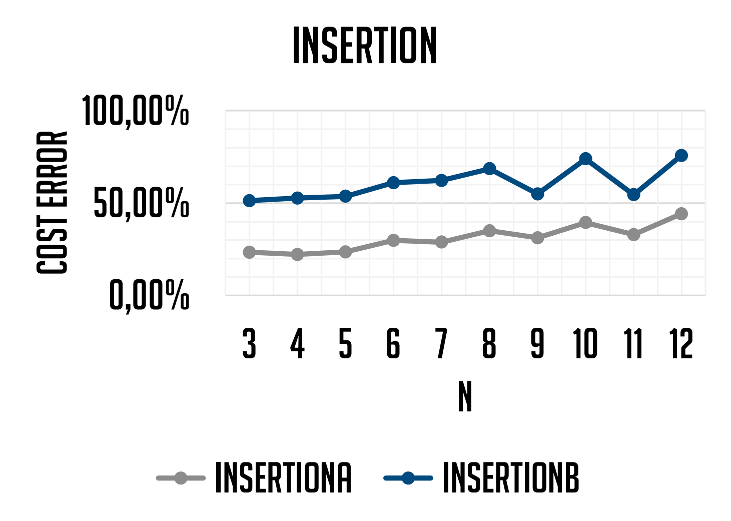


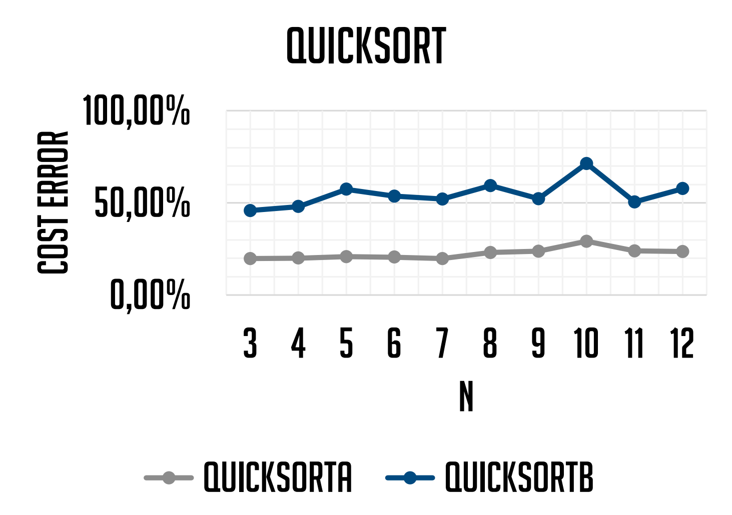


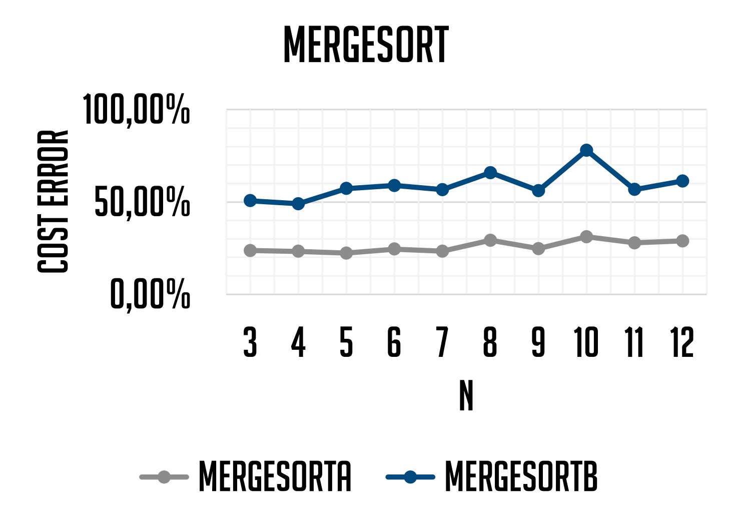
SORTING ALGORITHMS: COMPARISON A VS COMPARISON B


BUBBLE

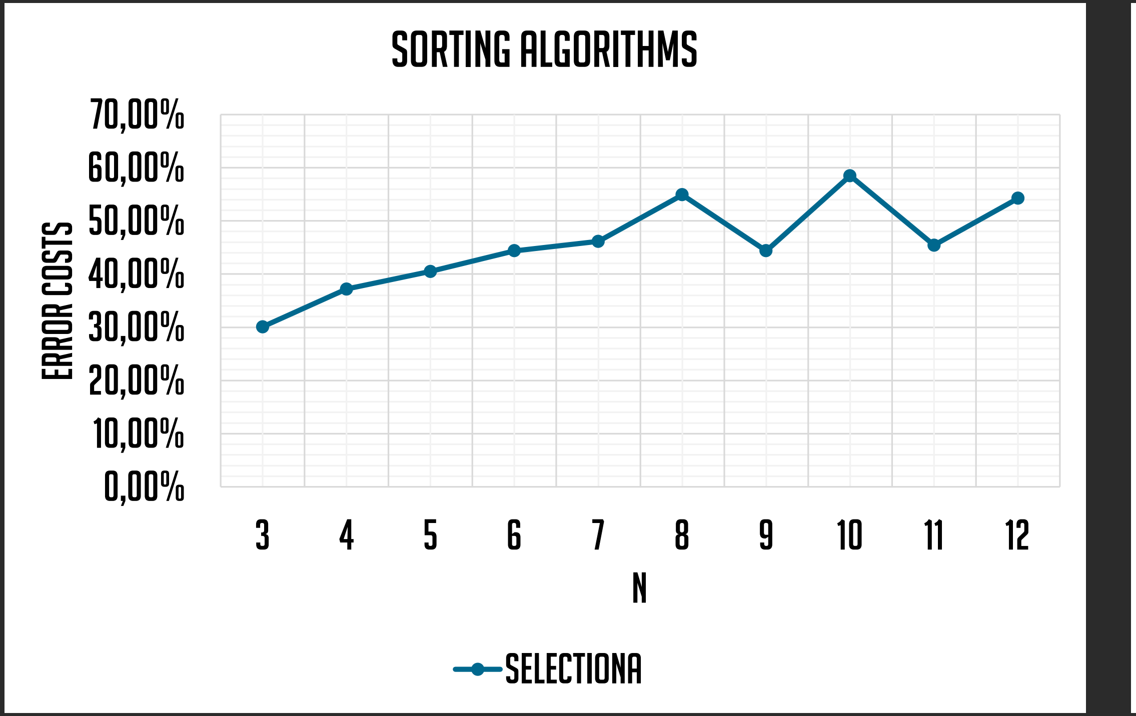

SELECTION

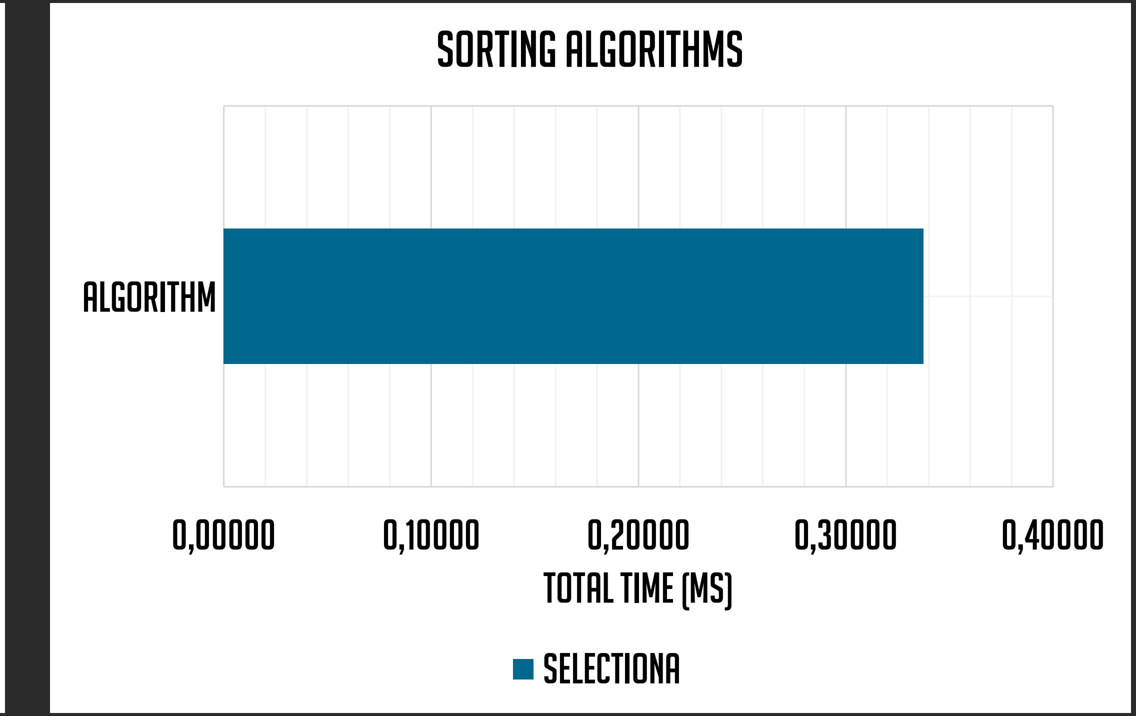

INSERTION


QUICKSORT


MERGESORT

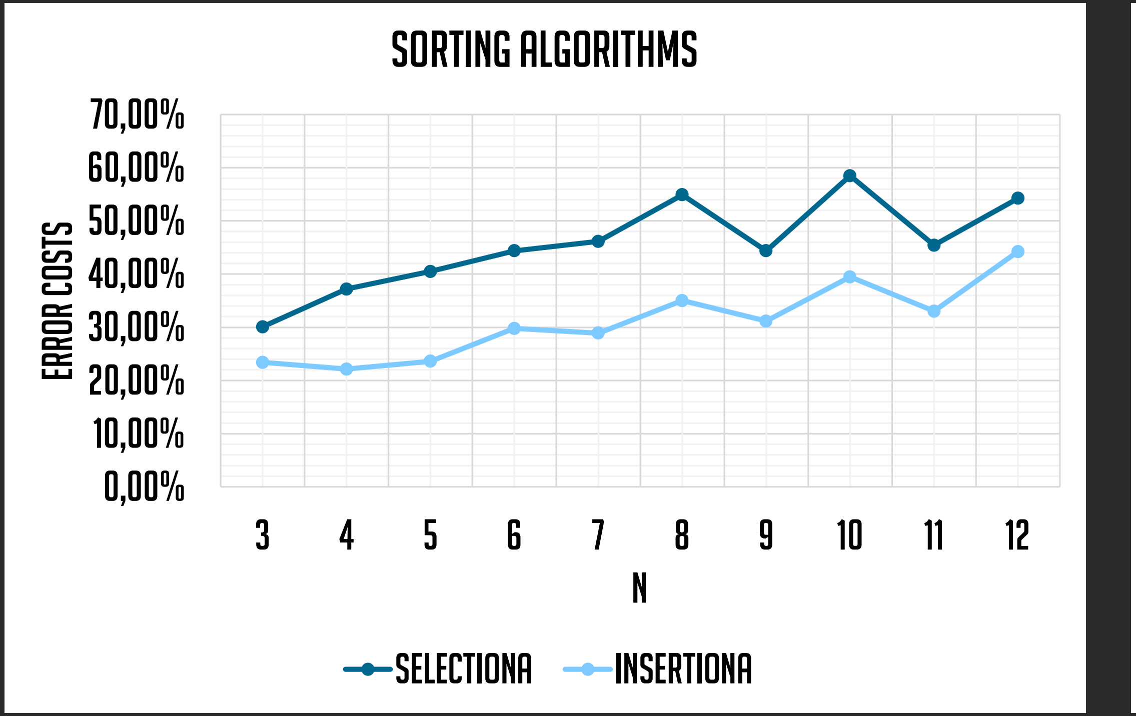
SORTING ALGORITHMS: GENERAL

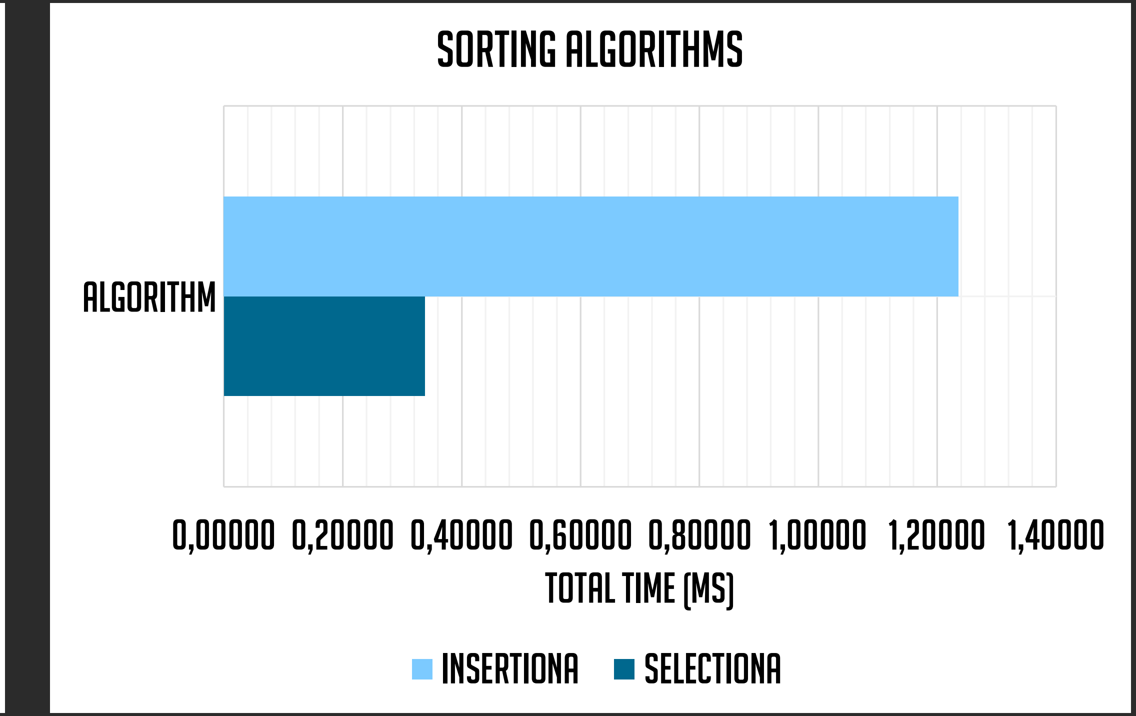


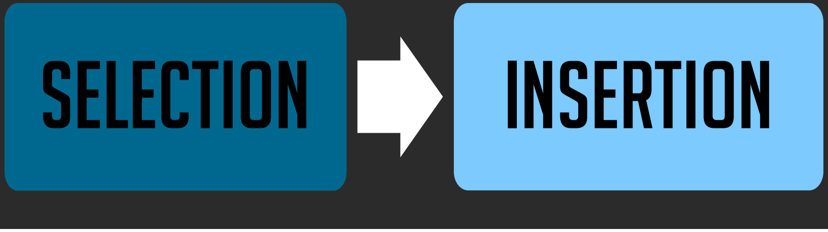


SELECTION

SORTING ALGORITHMS: GENERAL

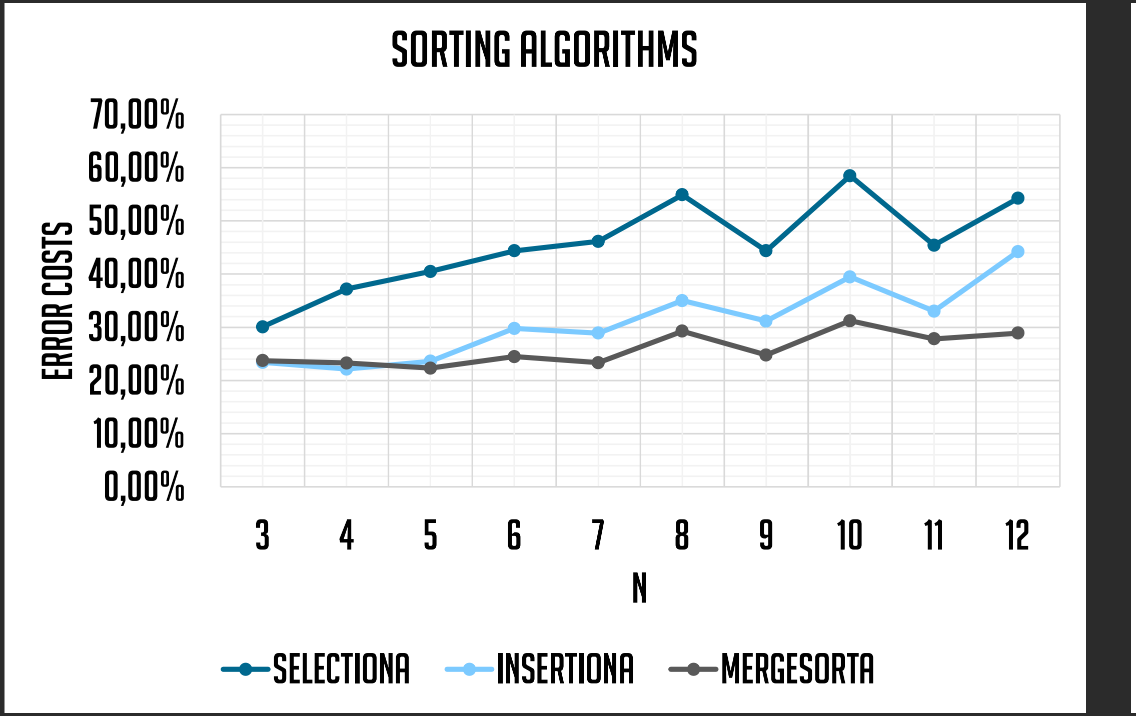


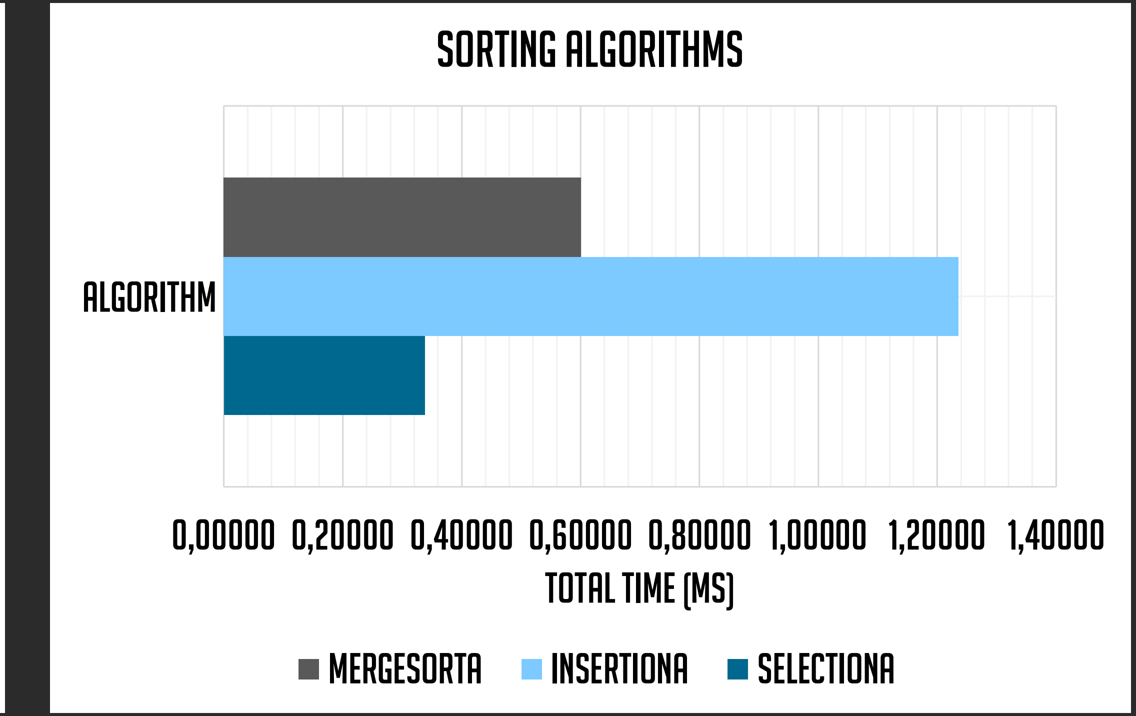



SELECTION

INSERTION

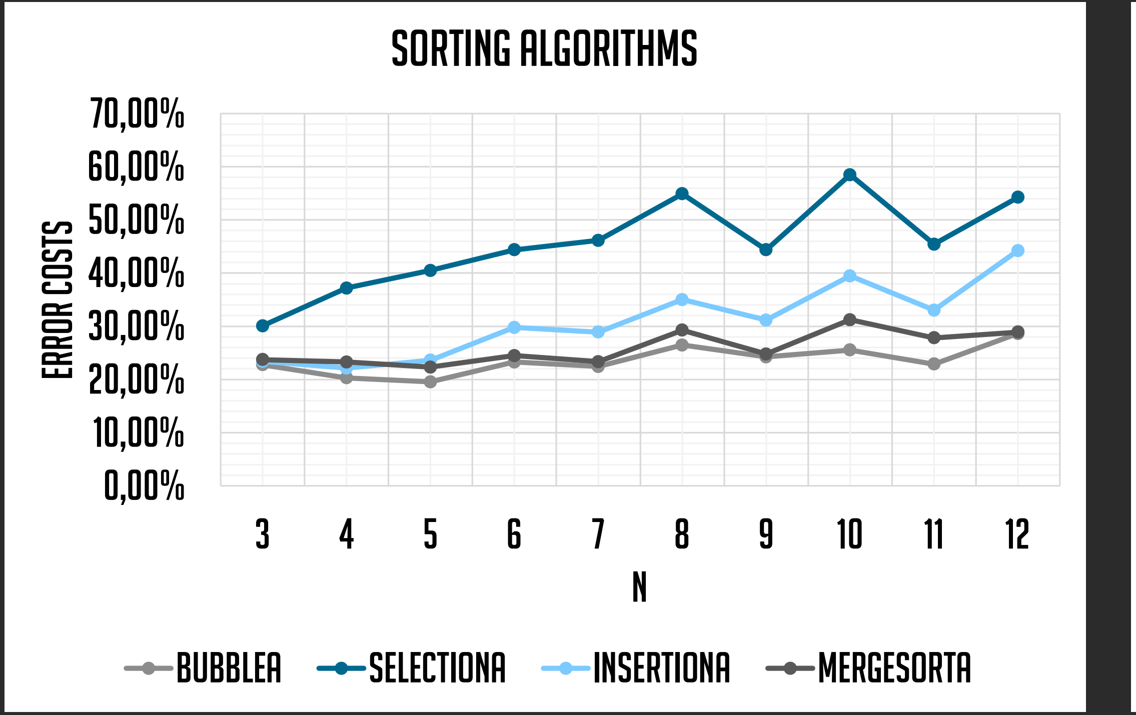
SORTING ALGORITHMS: GENERAL

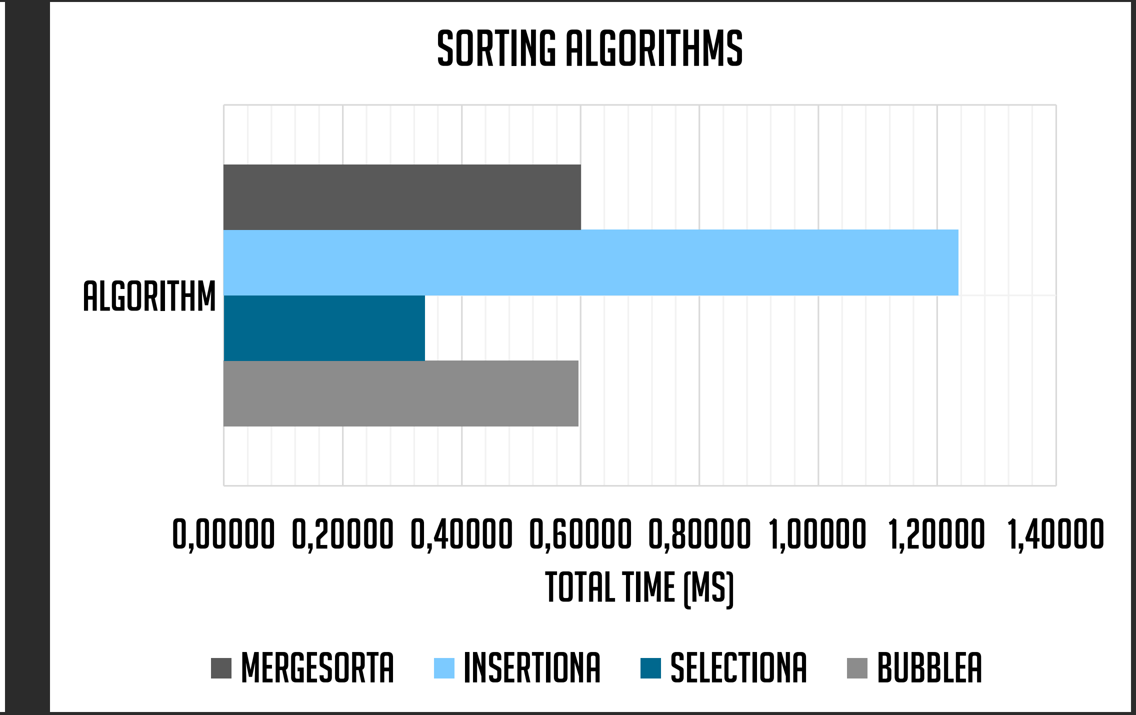


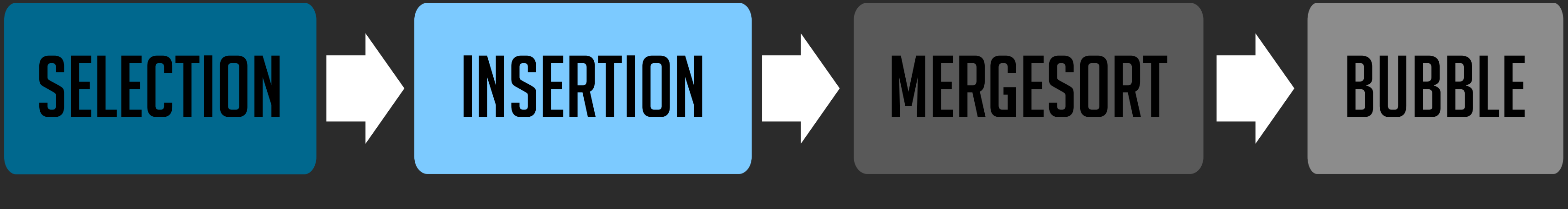


SELECTION → INSERTION → MERGESORT

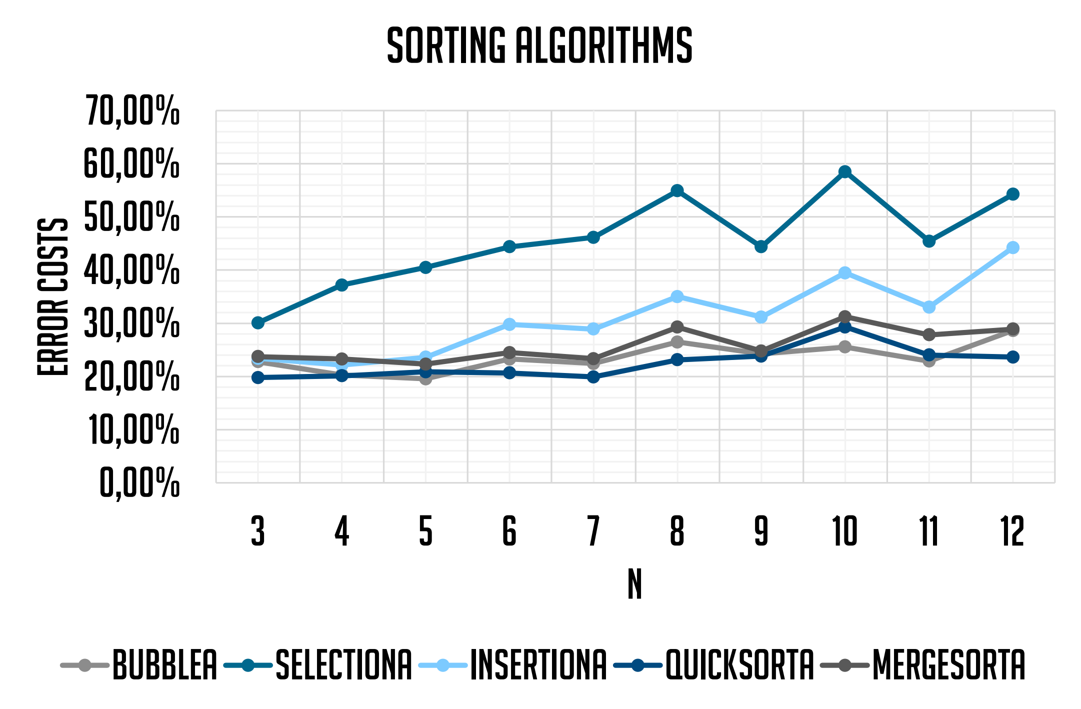
SORTING ALGORITHMS: GENERAL

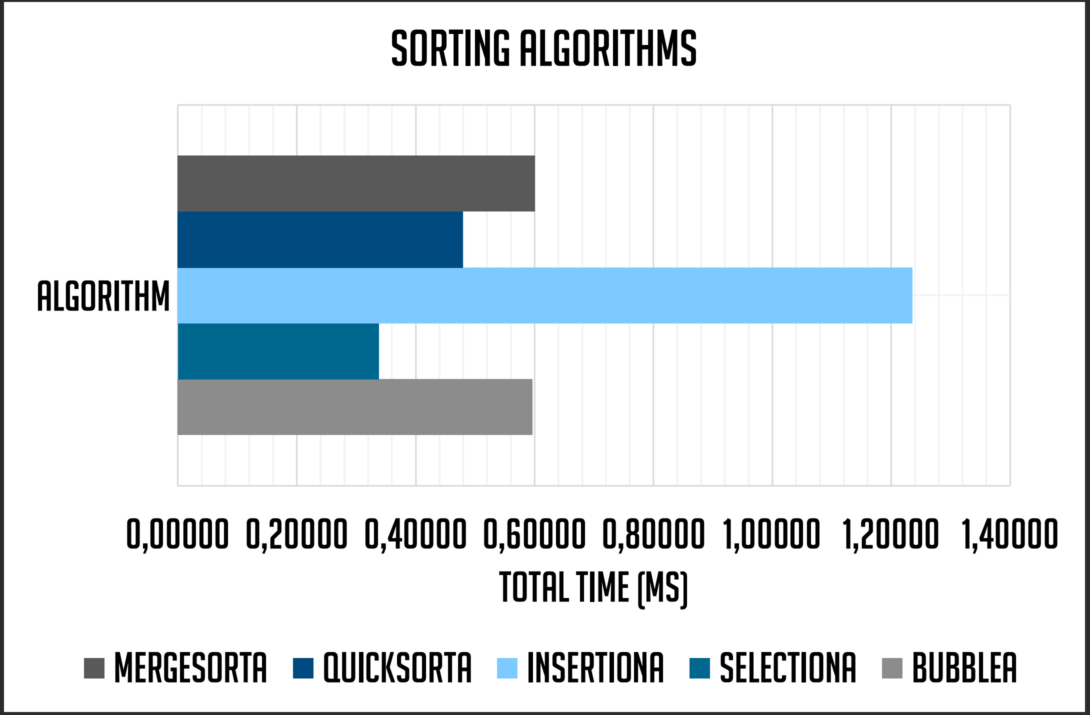


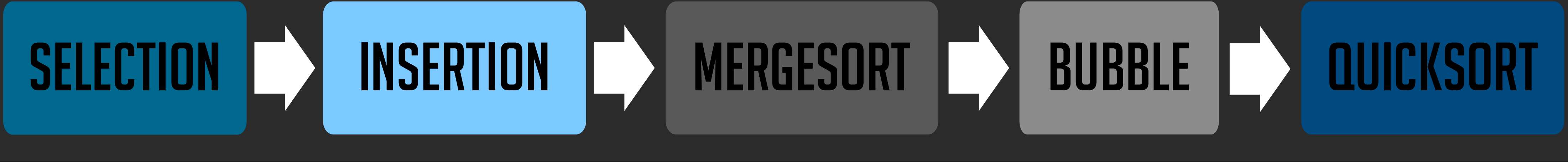




SORTING ALGORITHMS: GENERAL

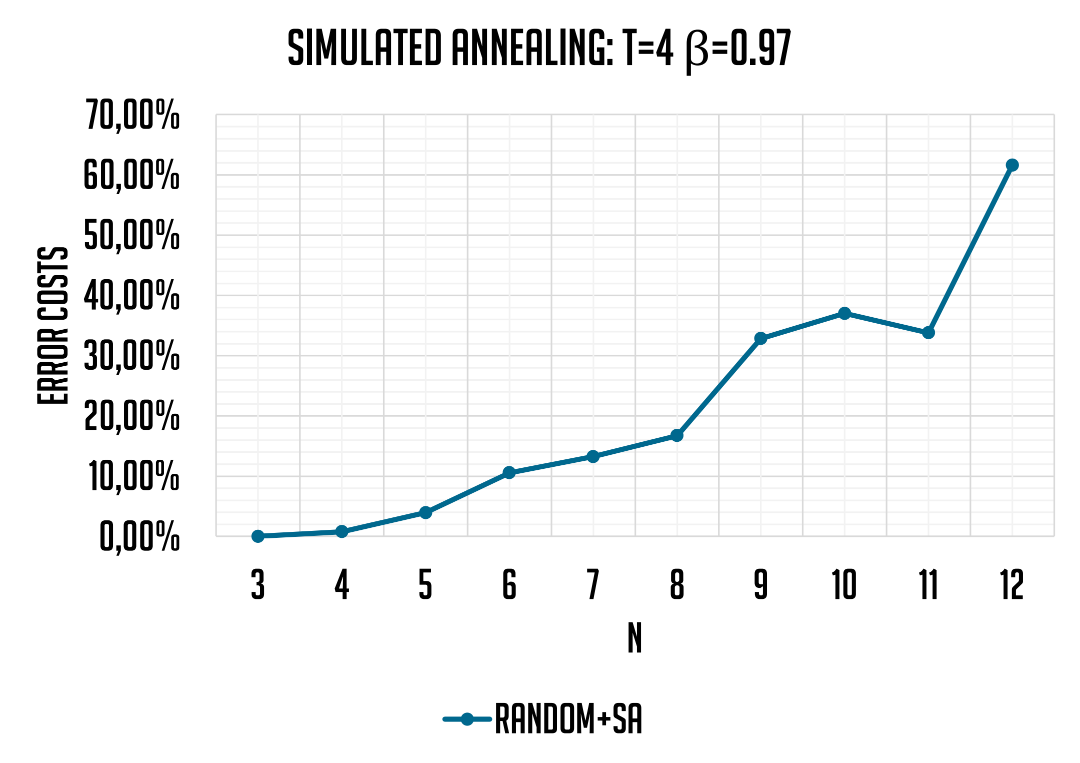


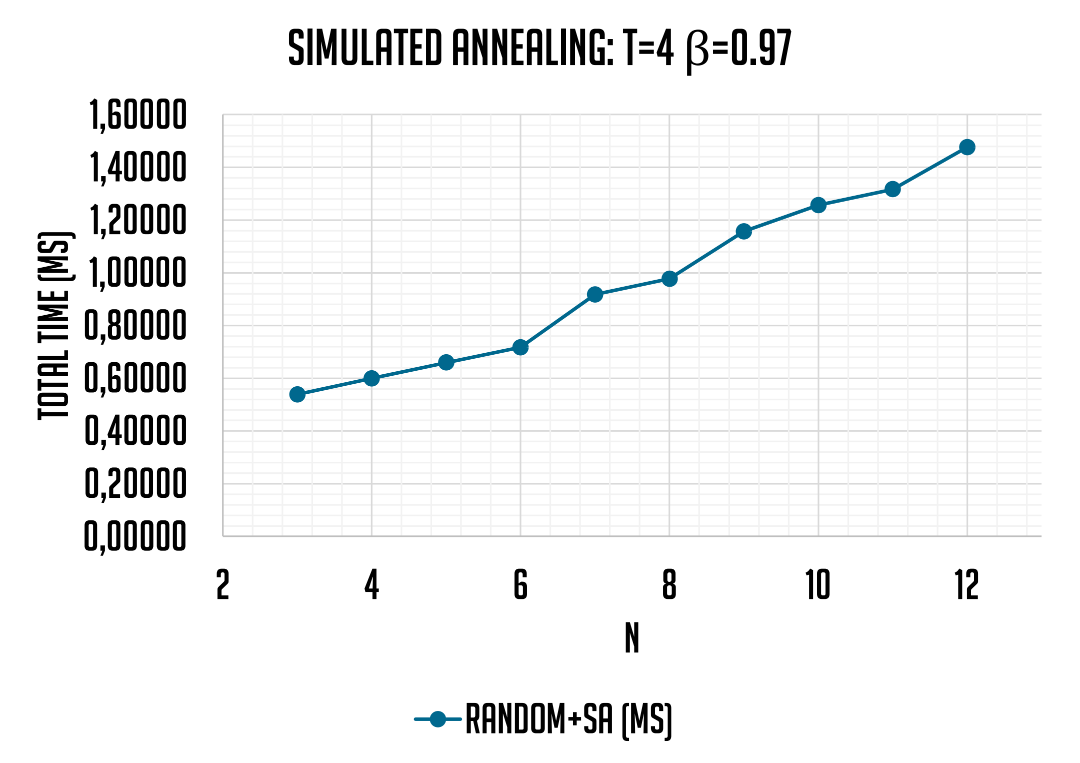




SELECTION → INSERTION → MERGESORT → BUBBLE → QUICKSORT

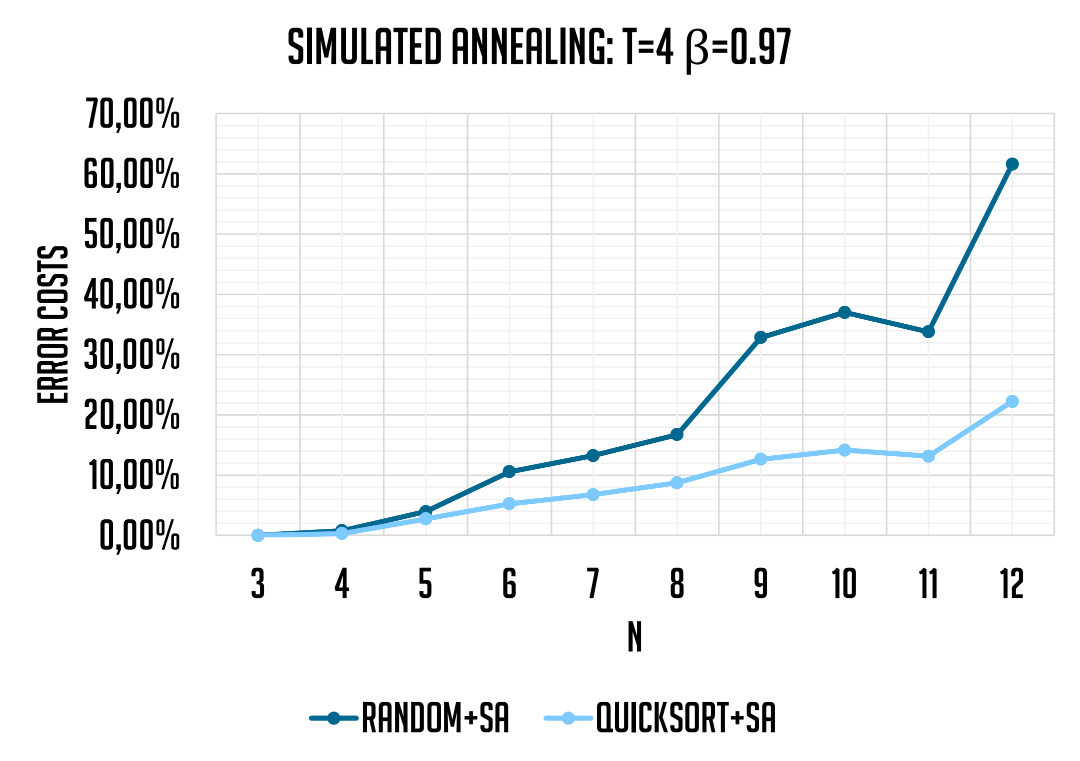
SIMULATED ANNEALING: INITIAL ALGORITHM

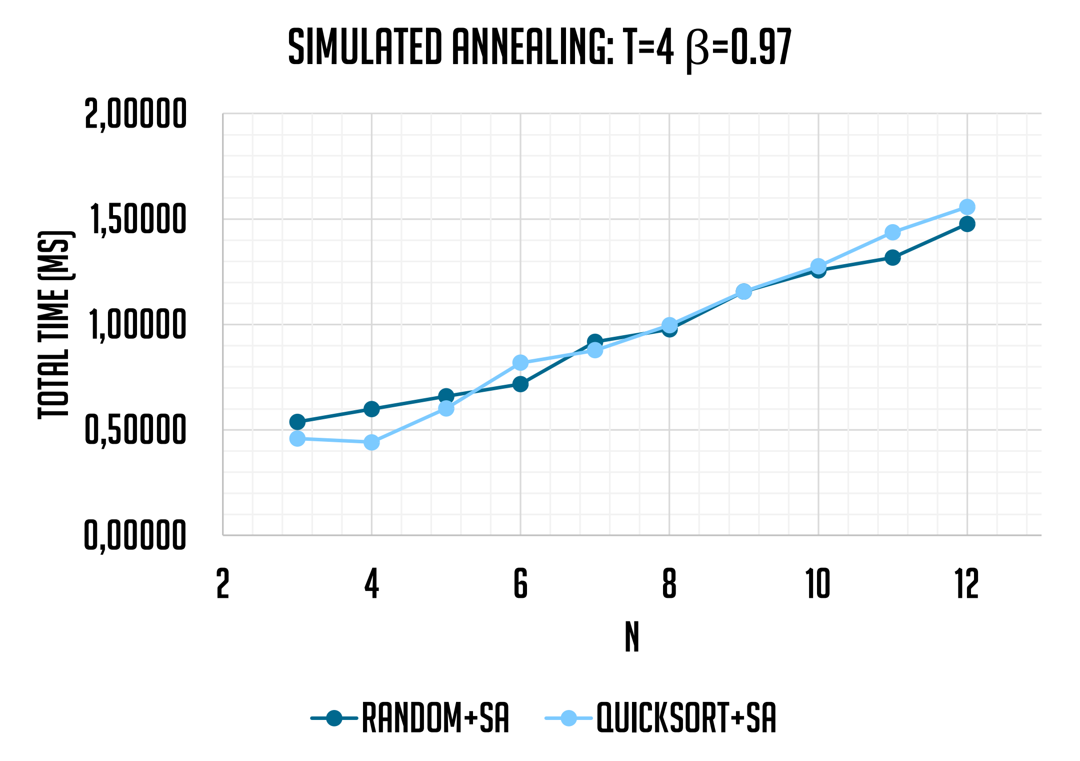


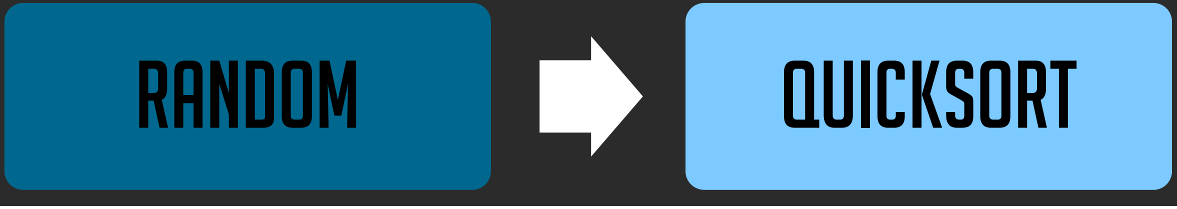


RANDOM

SIMULATED ANNEALING: INITIAL ALGORITHM

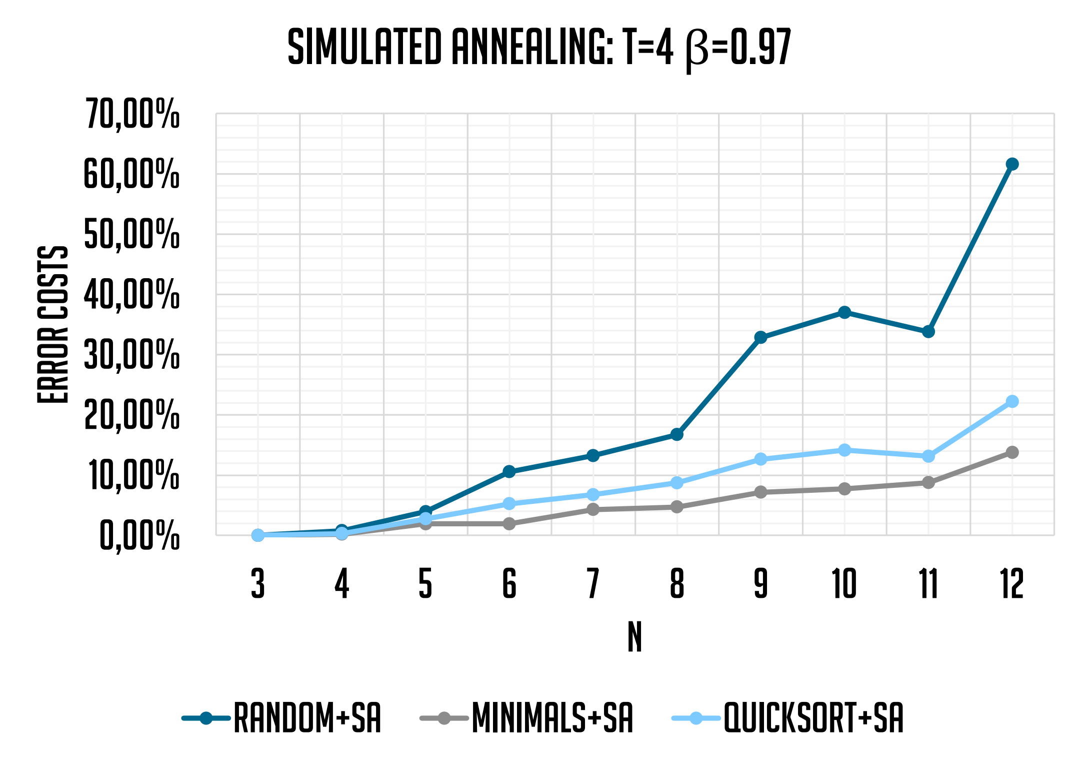

SIMULATED ANNEALING: T=4 β=0.97

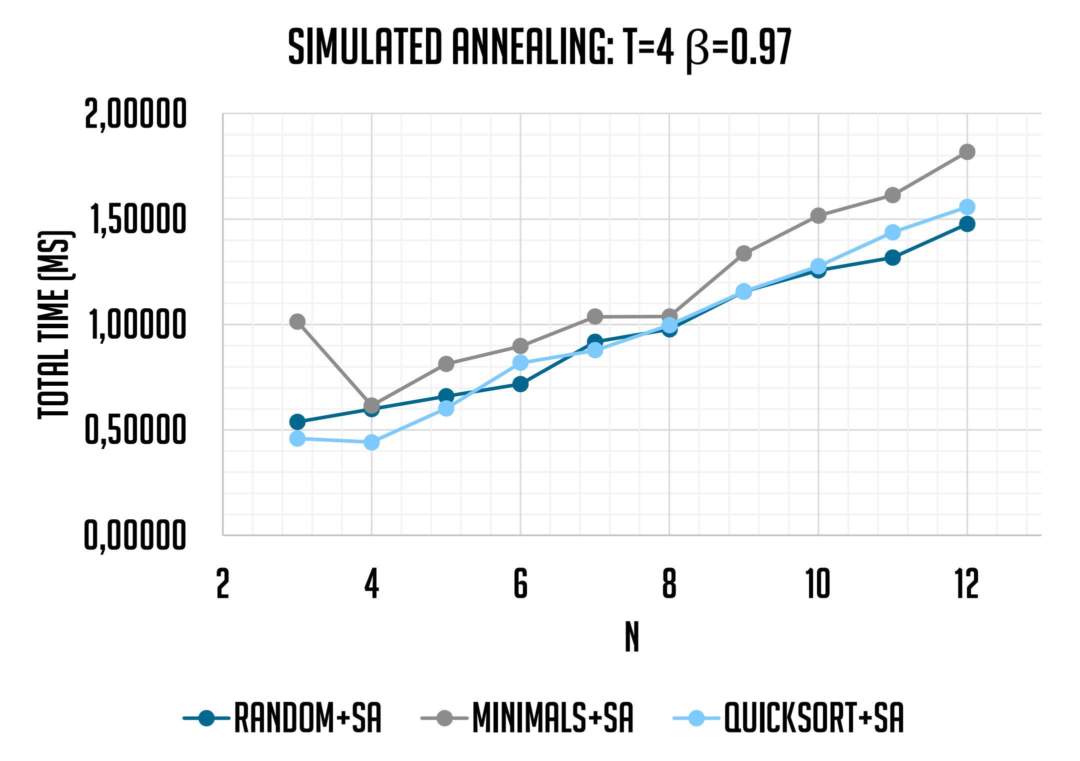

SIMULATED ANNEALING: T=4 β=0.97

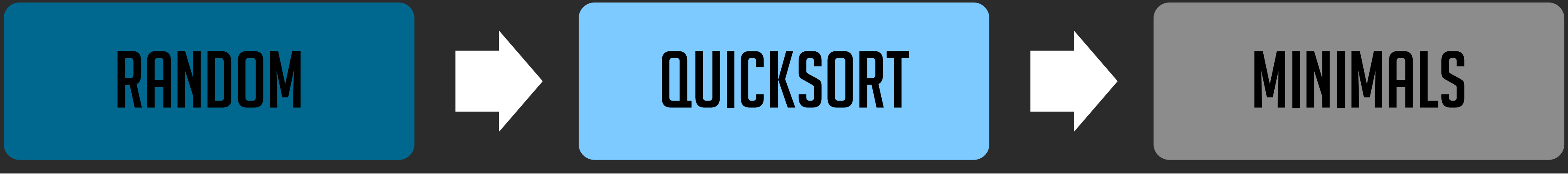

RANDOM

QUICKSORT

SIMULATED ANNEALING: INITIAL ALGORITHM

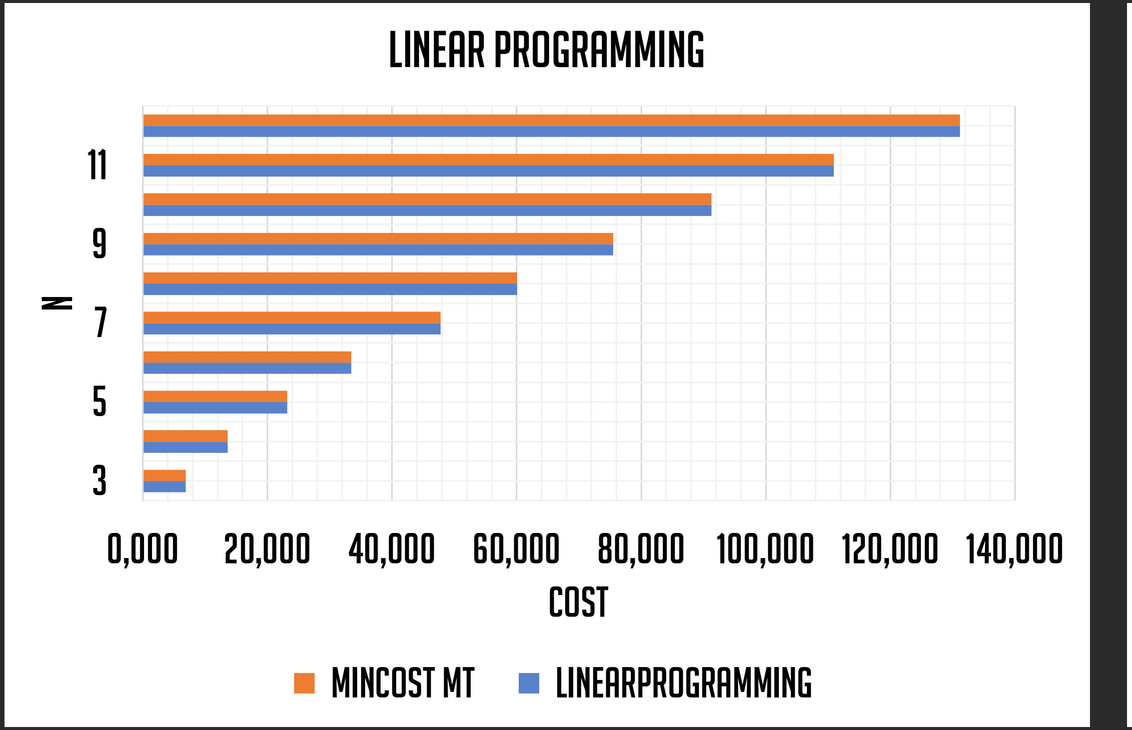


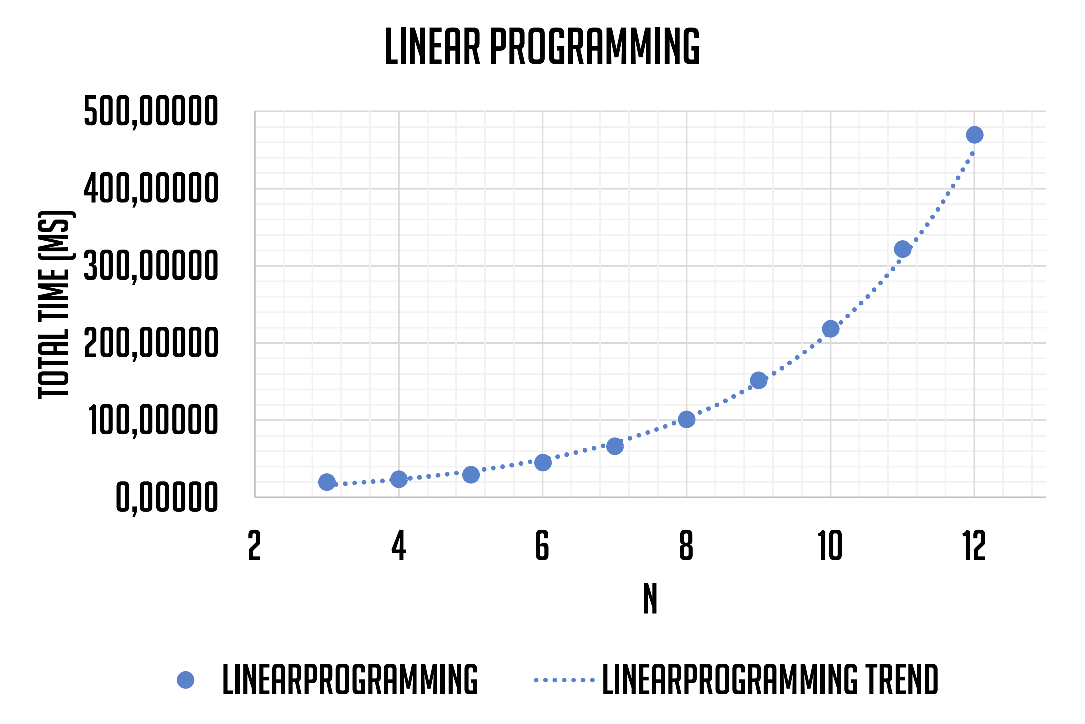




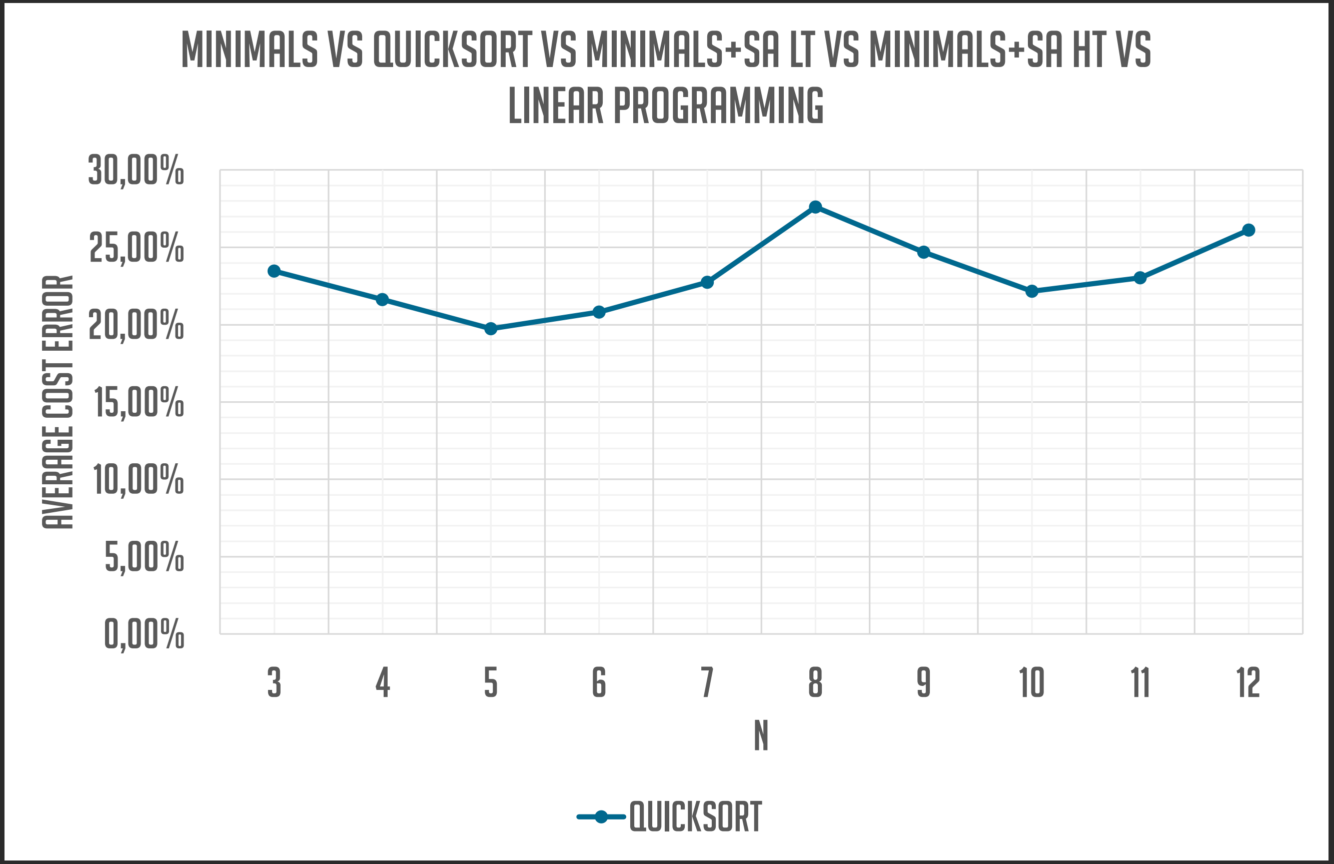
RANDOM → QUICKSORT → MINIMALS

LINEAR PROGRAMMING



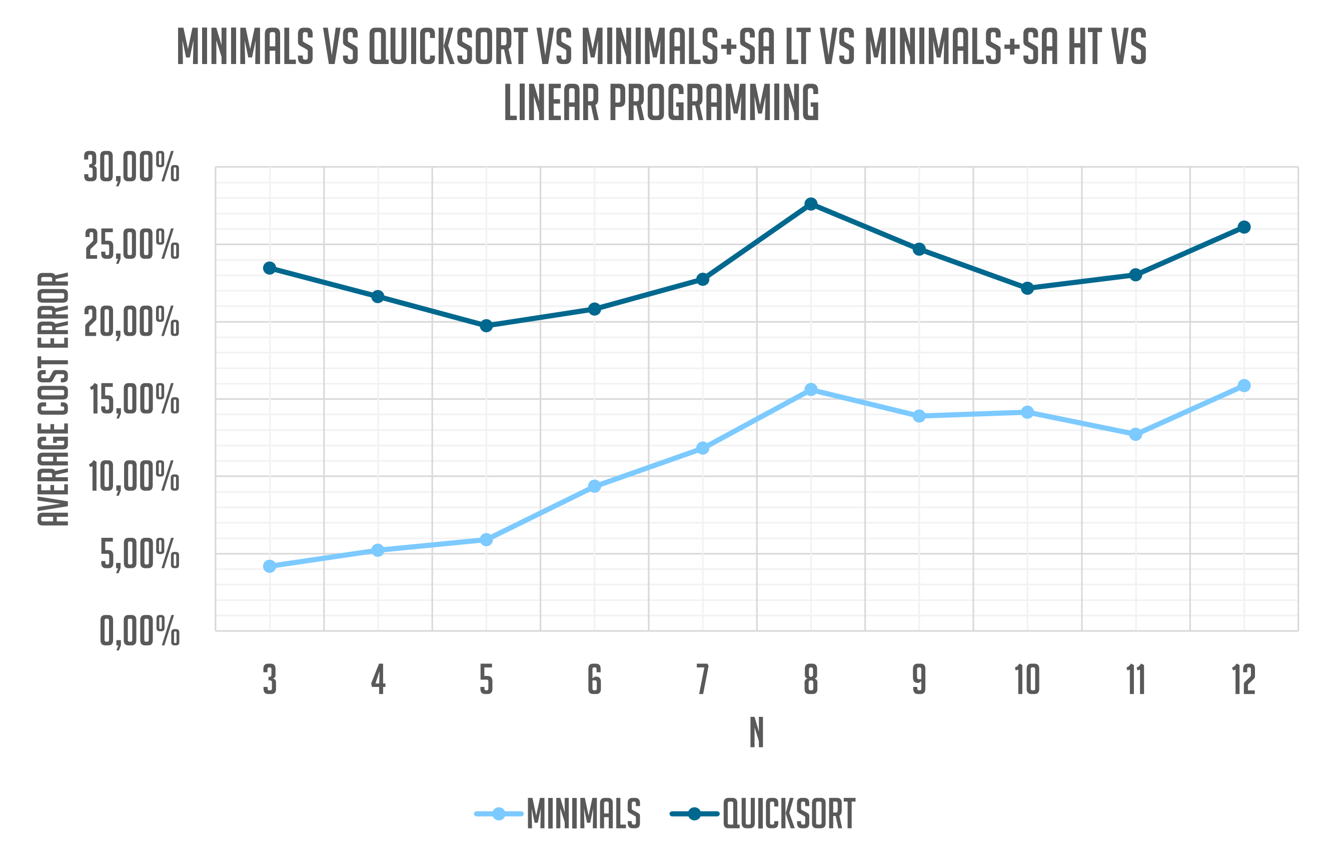


BEST AGGREGATION METHOD



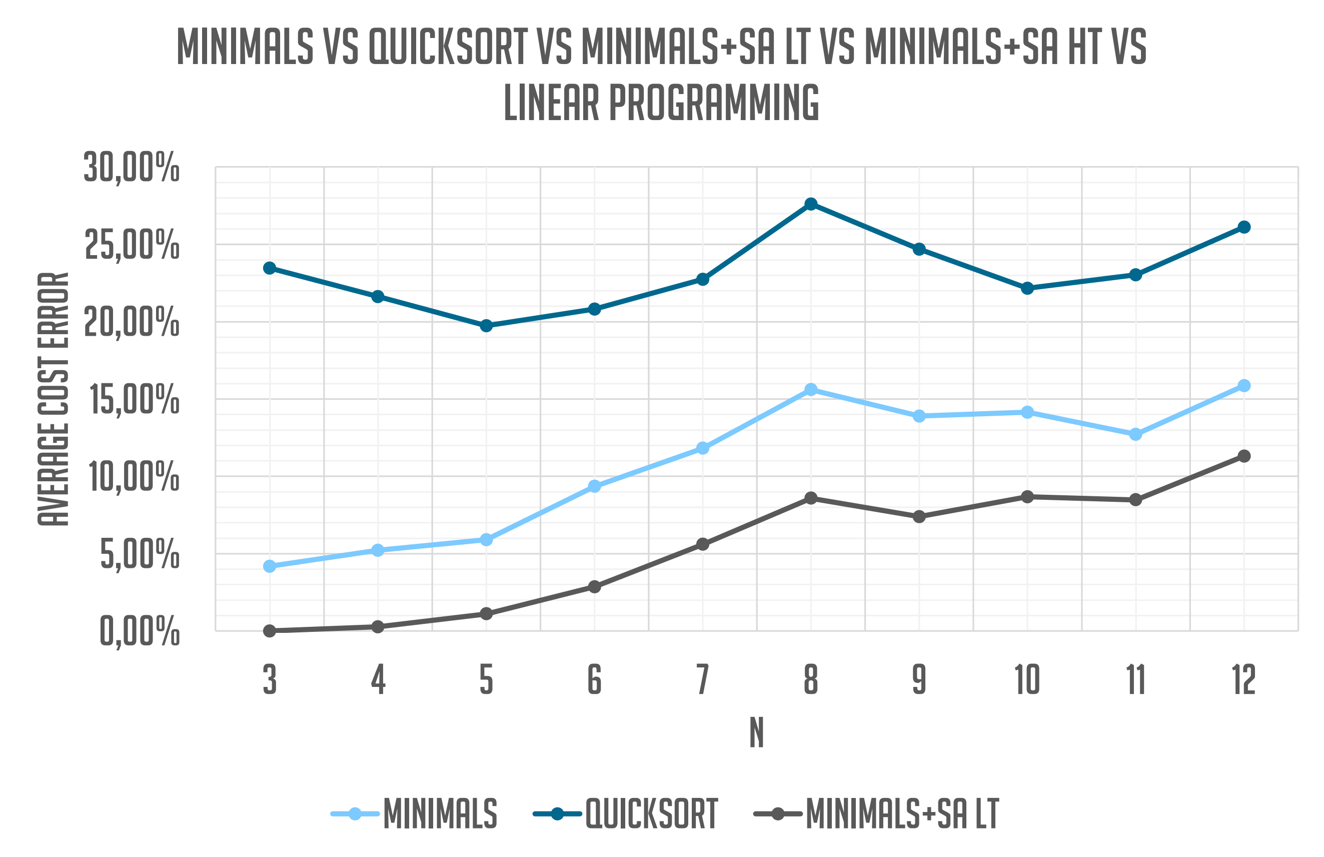
QUICKSORT

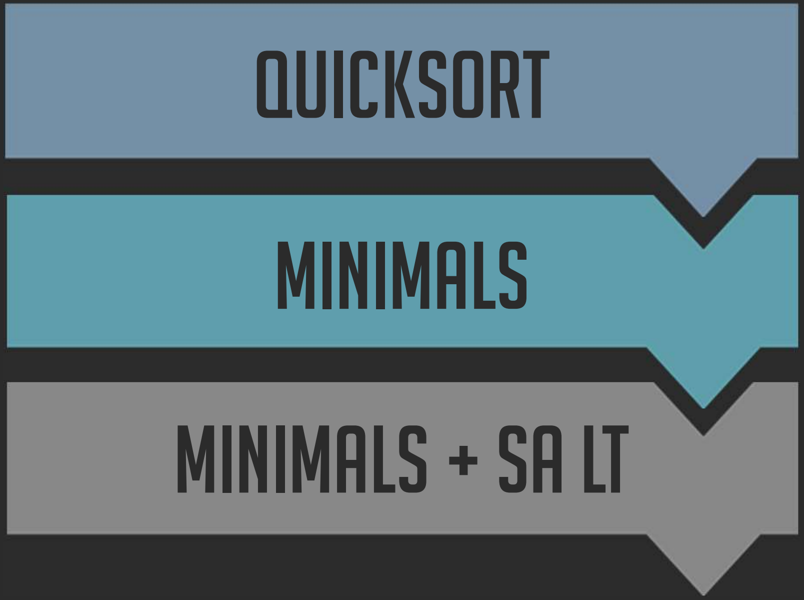
BEST AGGREGATION METHOD



QUICKSORT
MINIMALS

BEST AGGREGATION METHOD

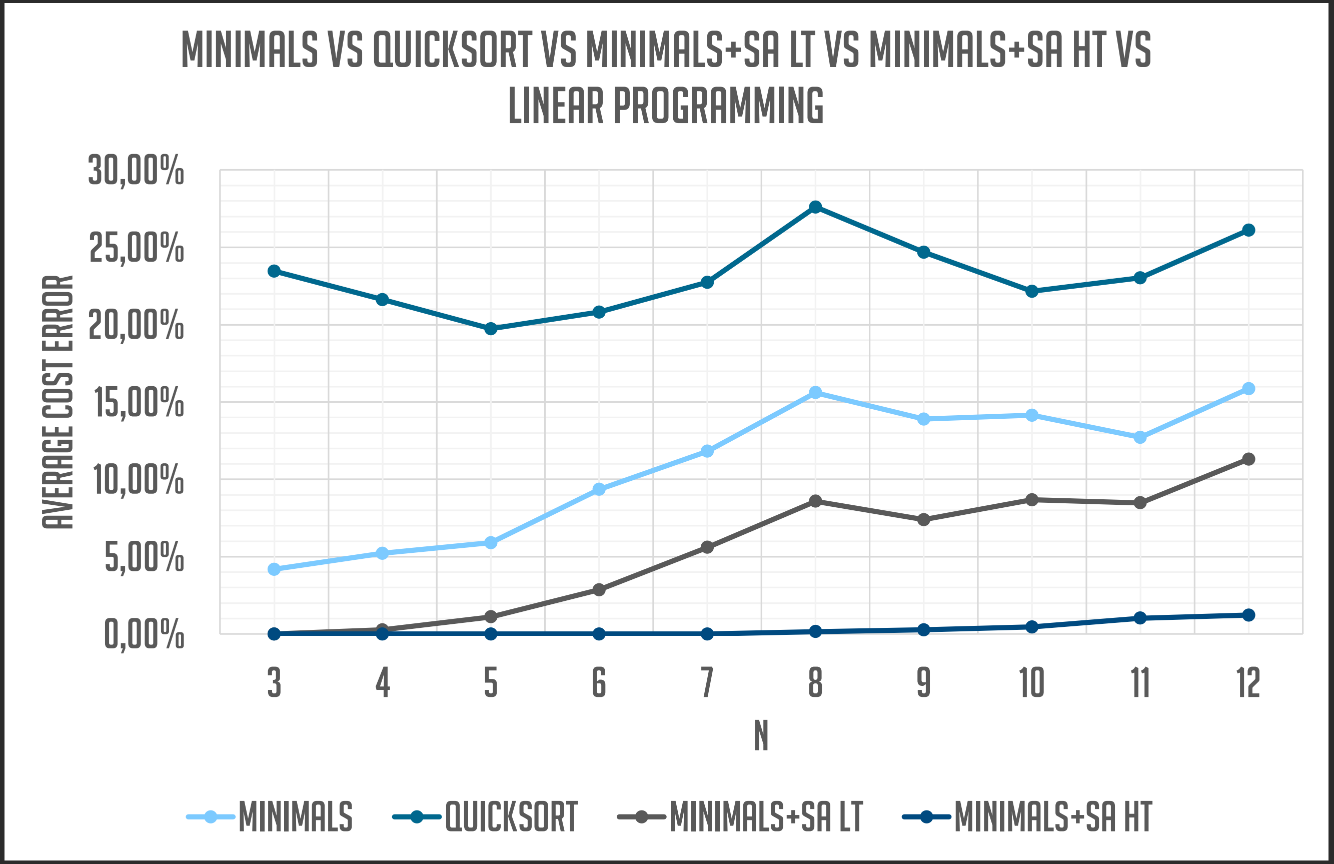

MINIMALS VS QUICKSORT VS MINIMALS+SA LT VS MINIMALS+SA HT VS LINEAR PROGRAMMING

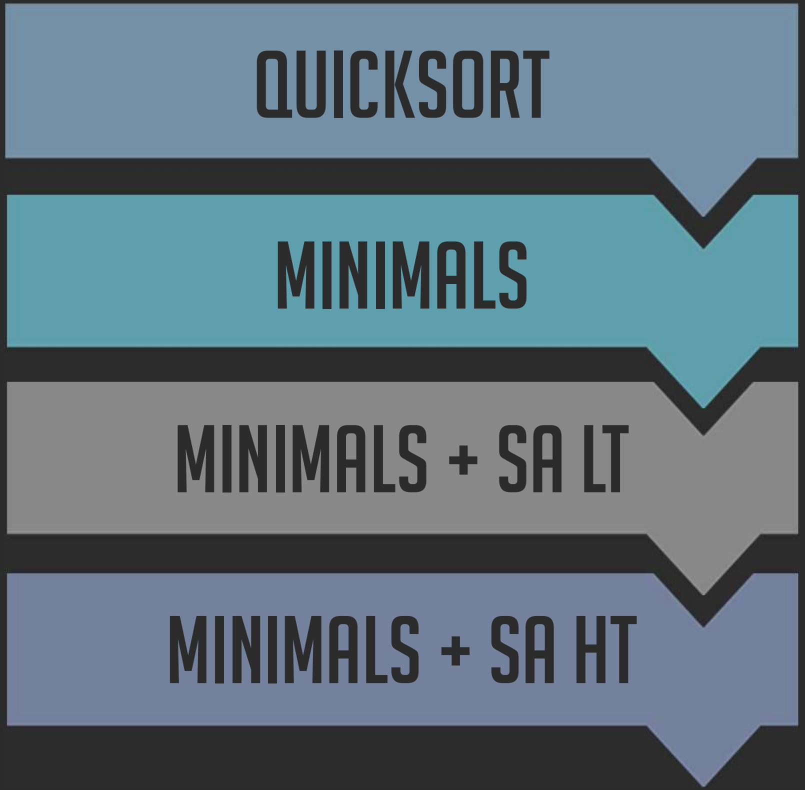

QUICKSORT

MINIMALS

MINIMALS + SA LT

BEST AGGREGATION METHOD



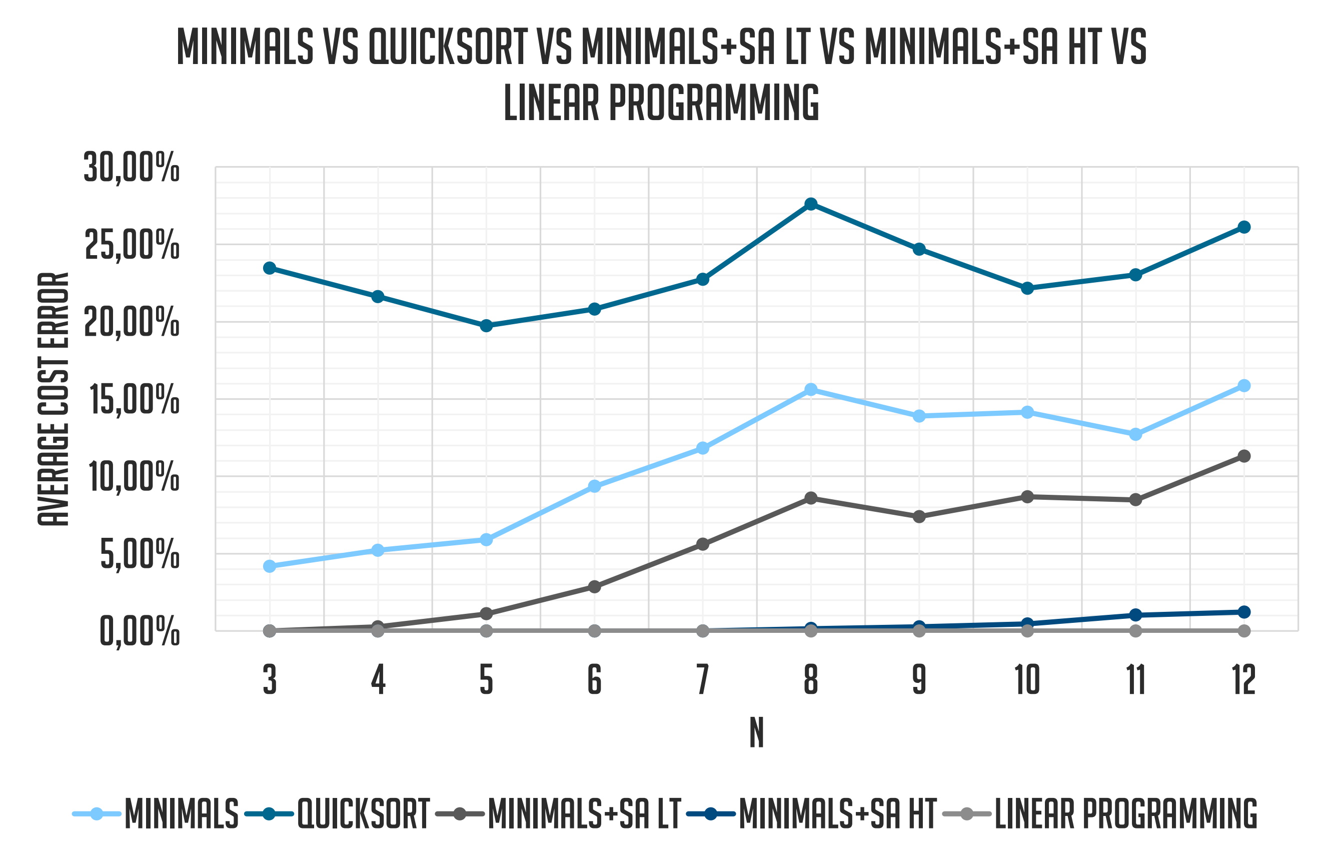

QUICKSORT

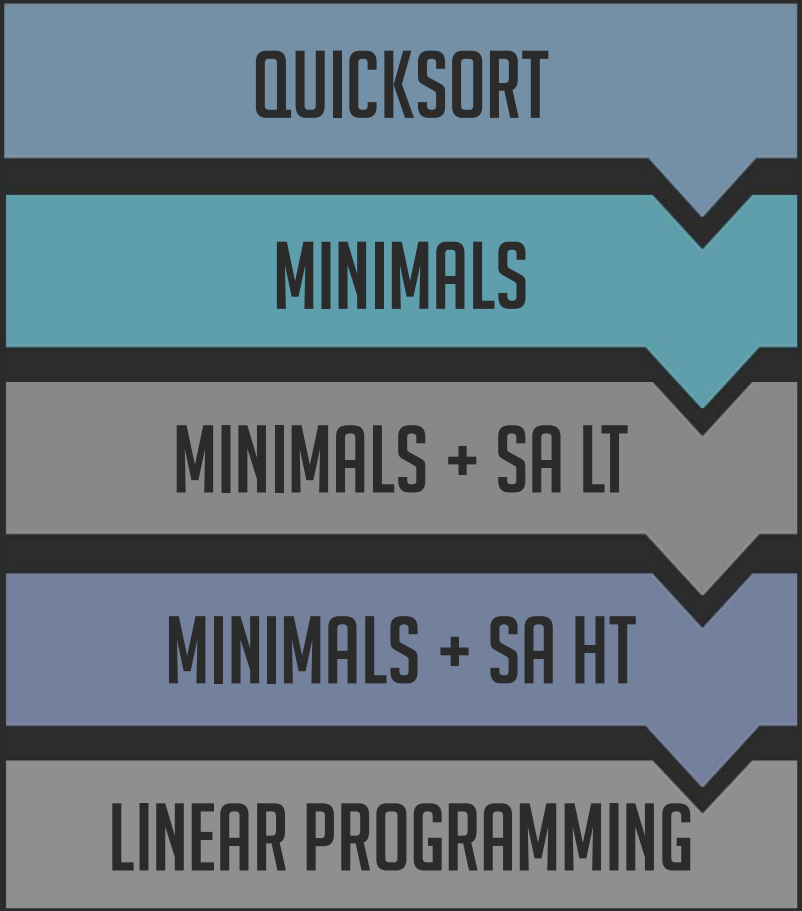
MINIMALS

MINIMALS + SA LT

MINIMALS + SA HT

BEST AGGREGATION METHOD





QUICKSORT

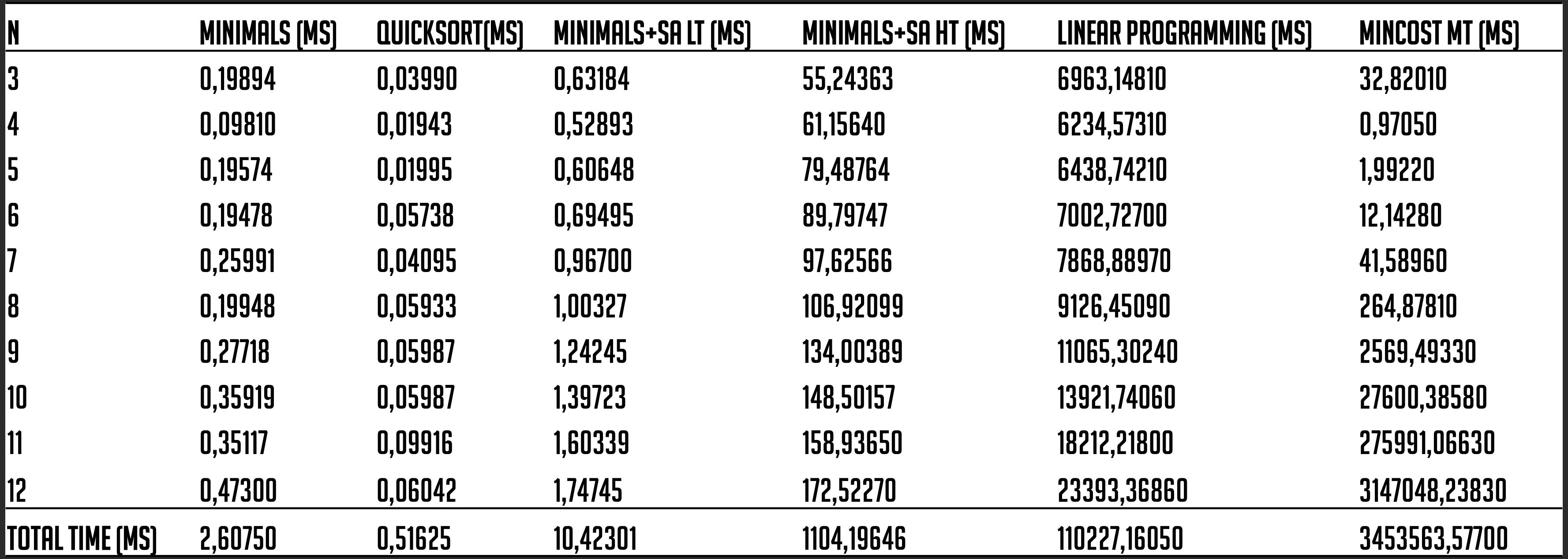
MINIMALS

MINIMALS + SA LT

MINIMALS + SA HT

LINEAR PROGRAMMING

BEST AGGREGATION METHOD



N	MINIMALS (MS)	QUICKSORT(MS)	MINIMALS+SA LT (MS)	MINIMALS+SA HT (MS)	LINEAR PROGRAMMING (MS)	MINCOST MT (MS)
3	0,19894	0,03990	0,63184	55,24363	6963,14810	32,82010
4	0,09810	0,01943	0,52893	61,15640	6234,57310	0,97050
5	0,19574	0,01995	0,60648	79,48764	6438,74210	1,99220
6	0,19478	0,05738	0,69495	89,79747	7002,72700	12,14280
7	0,25991	0,04095	0,96700	97,62566	7868,88970	41,58960
8	0,19948	0,05933	1,00327	106,92099	9126,45090	264,87810
9	0,27718	0,05987	1,24245	134,00389	11065,30240	2569,49330
10	0,35919	0,05987	1,39723	148,50157	13921,74060	27600,38580
11	0,35117	0,09916	1,60339	158,93650	18212,21800	275991,06630
12	0,47300	0,06042	1,74745	172,52270	23393,36860	3147048,23830
TOTAL TIME (MS)	2,60750	0,51625	10,42301	1104,19646	110227,16050	3453563,57700

CONCLUSIONS

SUMMARY

MINIMALS

MINCOST ST

MINIMALS
RANDOM

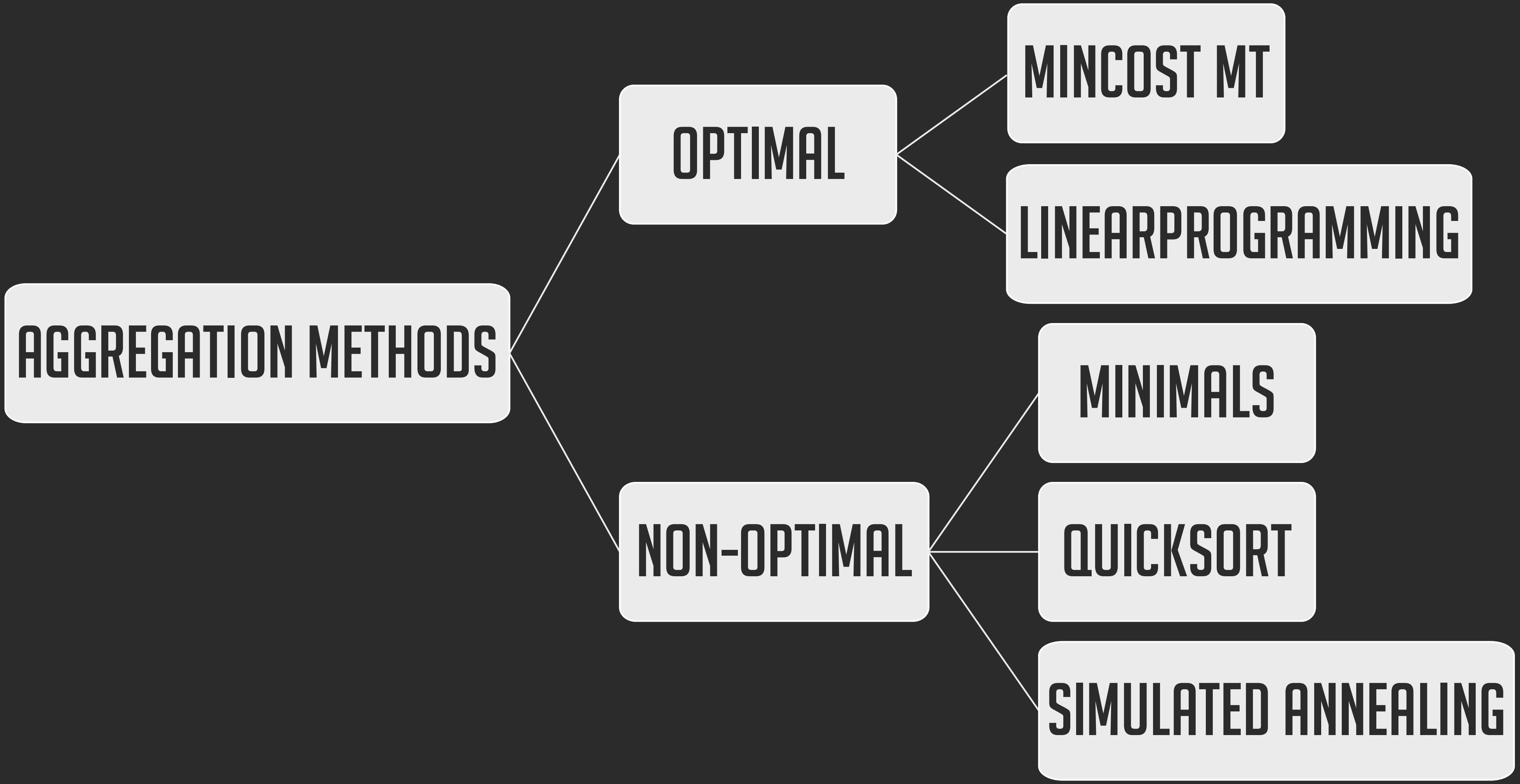
LINEAR
PROGRAMMING

MINCOST MT

SIMULATED
ANNEALING

SORTING
ALGORITHMS

TWO CATEGORIES OF ALGORITHMS


AGGREGATION METHODS
OPTIMAL
NON-OPTIMAL
MINCOST MT
LINEARPROGRAMMING
MINIMALS
QUICKSORT
SIMULATED ANNEALING

SIMULATED ANNEALING

QUALITY OF THE INITIAL SOLUTION

TEMPERATURE AND COOLING CONSTANT

OPTIMAL ALGORITHMS

MINCOST MT OR LINEARPROGRAMMING?

NON-OPTIMAL ALGORITHMS

MINIMALS + SIMULATED ANNEALING

WHAT IS THE BEST AGGREGATION METHOD?

$?

WHAT IS THE BEST AGGREGATION METHOD?

HIGH TEMPERATURE

MINIMALS + SIMULATED ANNEALING

LOW COOLING CONSTANT

?