Quadratic Assignment Problems in Cost Function Networks (extended version from CPAIOR 2026)

Guidio Sewa[1], Coralie Marchau[3], David Allouche[1],
Simon de Givry[1], George Katsirelos[2], and Thomas Schiex[1]

[1]INRAE, Toulouse, France

[2]INRAE, Paris-Saclay, France

[3]IMT Atlantique Nantes, France

RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité

INRAE

MIA TOULOUSE

ANITI
ARTIFICIAL & NATURAL INTELLIGENCE TOULOUSE INSTITUTE

Introduction

Contributions

- Adapted Singleton Consistency to Cost Function Networks, inspired by approaches from Integer Programming.
- Apply the resulting method on the Quadratic Assignment Problem.

- Find better solver parameter settings using irace configurator

Introduction

Contributions

- Adapted Singleton Consistency to Cost Function Networks, inspired by approaches from Integer Programming.
- Apply the resulting method on the Quadratic Assignment Problem.
- Find better solver parameter settings using irace configurator

Objective

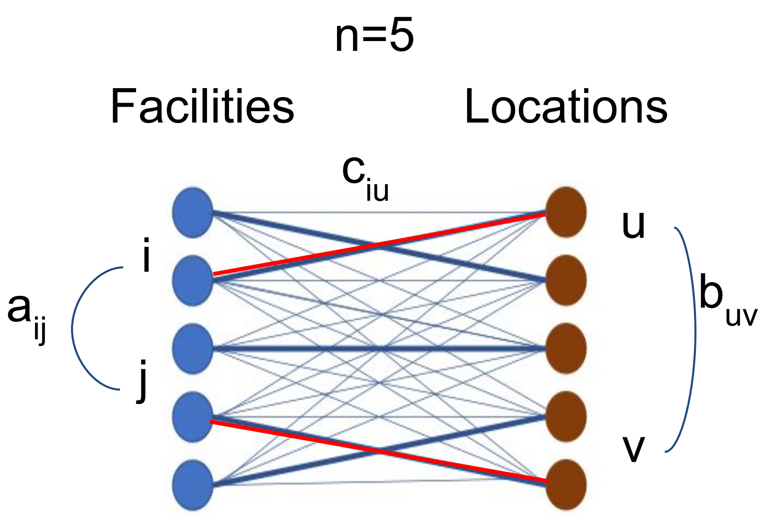
- Prune variable domains.
- Improve global lower bound of the problem during preprocessing.
- Automatic solver parameter configuration.

Quadratic Assignment Problem

(Koopmans & Beckmann. Journal of the Econometric Society, 1957)

Find an assignment of size n minimizing a quadratic objective function

Bipartite Graph Representation



Quadratic Assignment Problem

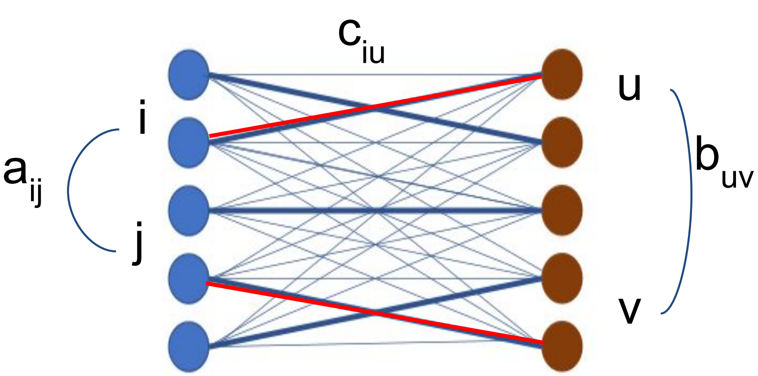
(Koopmans & Beckmann. Journal of the Econometric Society, 1957)

Find an assignment of size n minimizing a quadratic objective function

Bipartite Graph Representation

n=5

Facilities Locations



Integer Programming Formulation :

$$q = \min \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{u=1}^{n} \sum_{v=1}^{n} a_{ij} b_{uv} x_{iu} x_{jv} + \sum_{i=1}^{n} \sum_{u=1}^{n} c_{iu} x_{iu}$$

$$\sum_{u=1}^{n} x_{iu} = 1 \quad \forall i \in \{1, \ldots, n\}$$

$$\sum_{i=1}^{n} x_{iu} = 1 \quad \forall u \in \{1, \ldots, n\}$$

$$x_{iu} \in \{0, 1\} \quad \forall i, u \in \{1, \ldots, n\}^2$$

Quadratic Assignment Problem

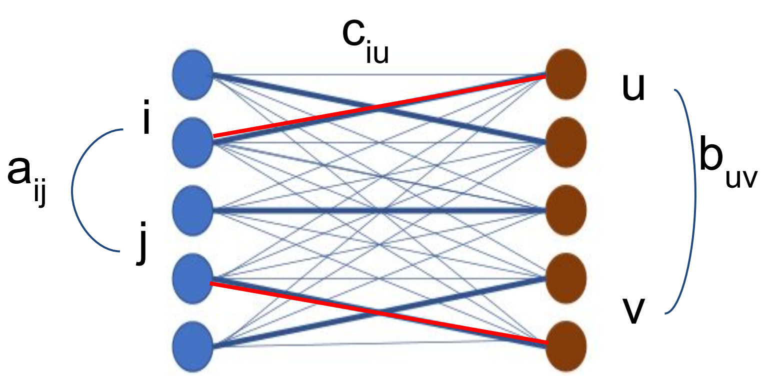
(Koopmans & Beckmann. Journal of the Econometric Society, 1957)

Find an assignment of size n minimizing a quadratic objective function

Bipartite Graph Representation

n=5

Facilities Locations



Integer Programming Formulation :

$$q = \min \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{u=1}^{n}\sum_{v=1}^{n} a_{ij} b_{uv} x_{iu} x_{jv} + \sum_{i=1}^{n}\sum_{u=1}^{n} c_{iu} x_{iu}$$

$$\sum_{u=1}^{n} x_{iu} = 1 \quad \forall i \in \{1, \ldots, n\}$$

$$\sum_{i=1}^{n} x_{iu} = 1 \quad \forall u \in \{1, \ldots, n\}$$

$$x_{iu} \in \{0, 1\} \quad \forall i, u \in \{1, \ldots, n\}^2$$

flow matrix a_{ij}

Quadratic Assignment Problem

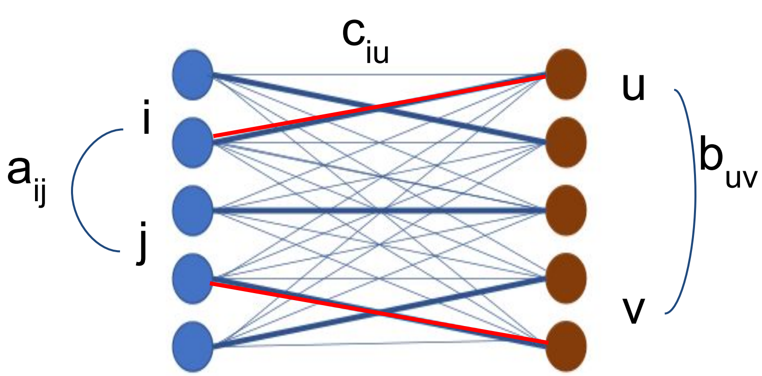
(Koopmans & Beckmann. Journal of the Econometric Society, 1957)

Find an assignment of size n minimizing a quadratic objective function

Bipartite Graph Representation

n=5

Facilities Locations



Integer Programming Formulation :

$$q = \min \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{u=1}^{n} \sum_{v=1}^{n} a_{ij} b_{uv} x_{iu} x_{jv} + \sum_{i=1}^{n} \sum_{u=1}^{n} c_{iu} x_{iu}$$

$$\sum_{u=1}^{n} x_{iu} = 1 \quad \forall i \in \{1, \ldots, n\}$$

$$\sum_{i=1}^{n} x_{iu} = 1 \quad \forall u \in \{1, \ldots, n\}$$

$$x_{iu} \in \{0, 1\} \quad \forall i, u \in \{1, \ldots, n\}^2$$

flow matrix a_{ij}

distance matrix b_{uv}

Quadratic Assignment Problem

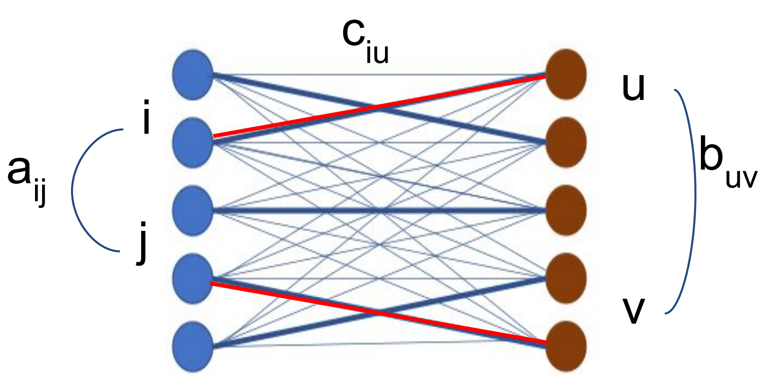
(Koopmans & Beckmann. Journal of the Econometric Society, 1957)

Find an assignment of size n minimizing a quadratic objective function

Bipartite Graph Representation

n=5

Facilities Locations



Integer Programming Formulation :

linear cost matrix c_{iu}

$$q = \min \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{u=1}^{n} \sum_{v=1}^{n} a_{ij} b_{uv} x_{iu} x_{jv} + \sum_{i=1}^{n} \sum_{u=1}^{n} c_{iu} x_{iu}$$

$$\sum_{u=1}^{n} x_{iu} = 1 \quad \forall i \in \{1, \ldots, n\}$$

$$\sum_{i=1}^{n} x_{iu} = 1 \quad \forall u \in \{1, \ldots, n\}$$

$$x_{iu} \in \{0, 1\} \quad \forall i, u \in \{1, \ldots, n\}^2$$

flow matrix a_{ij}

distance matrix b_{uv}

hypotheses: $a_{ii}=b_{uu}=0$; a_{ij}, b_{uv}, c_{iu} nonnegative integers

Quadratic Assignment Problem

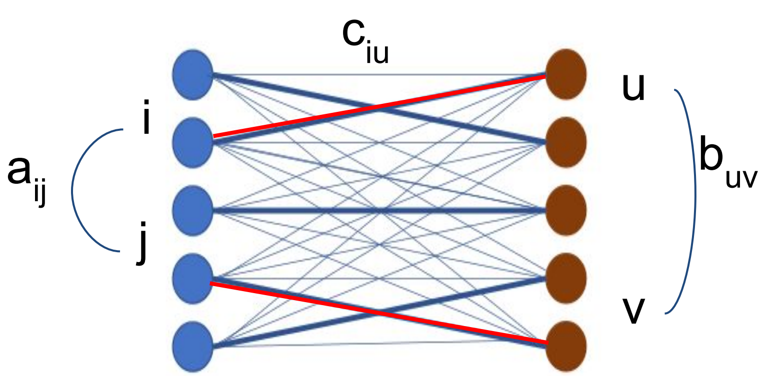
(Koopmans & Beckmann. Journal of the Econometric Society, 1957)

Find an assignment of size n minimizing a quadratic objective function

Bipartite Graph Representation

n=5

Facilities Locations



NP-hard problem

Integer Programming Formulation :

linear cost matrix c_{iu}

$$q = \min \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{u=1}^{n}\sum_{v=1}^{n} a_{ij} b_{uv} x_{iu} x_{jv} + \sum_{i=1}^{n}\sum_{u=1}^{n} c_{iu} x_{iu}$$

$$\sum_{u=1}^{n} x_{iu} = 1 \quad \forall i \in \{1, \ldots, n\}$$

$$\sum_{i=1}^{n} x_{iu} = 1 \quad \forall u \in \{1, \ldots, n\}$$

$$x_{iu} \in \{0, 1\} \quad \forall i, u \in \{1, \ldots, n\}^2$$

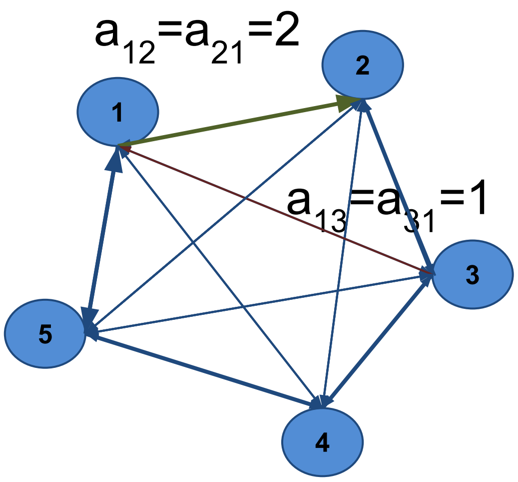
flow matrix a_{ij}

distance matrix b_{uv}

hypotheses: $a_{ii}=b_{uu}=0$; a_{ij}, b_{uv}, c_{iu} nonnegative integers

Quadratic Assignment Problem

Example with a dense flow matrix a_{ij}



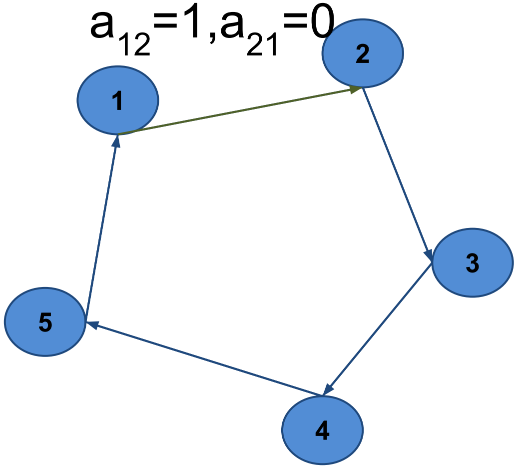
0	2	1	1	2
2	0	2	1	1
1	2	0	2	1
1	1	2	0	2
2	1	1	2	0

symmetric case for matrix a_{ij}

Quadratic Assignment Problem

Example with a sparse flow matrix a_{ij}

$a_{12}=1, a_{21}=0$



0	1	0	0	0
0	0	1	0	0
0	0	0	1	0
0	0	0	0	1
1	0	0	0	0

nonsymmetric case for matrix a_{ij}

Special case: Traveling Salesman Problem
(b_{uv} = distance between cities u and v)

Quadratic Assignment Problem

QAP Solving Methods

Exact Methods	Heuristic Methods
Branch and Cut (linearization, relaxation) CPLEX, Gurobi Constraint Programming (local consistency) CP-SAT, **Toulbar2**	Local Search (simulated annealing, tabu search) Hexaly

Gilmore Lawler Bound (GLB) for QAP

(Gilmore, Journal of the society for industrial and applied mathematics, 1962)

Gilmore bound consists of two steps :

Step 1 : solves n^2 linear assignment problems (LAP)

Complete problem

Subproblem with x_{iu}=1

$$q_1 = \min \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{u=1}^{n} \sum_{v=1}^{n} a_{ij} b_{uv} x_{iu} x_{jv}$$

$$\sum_{u=1}^{n} x_{iu} = 1 \quad \forall i \in \{1, \ldots, n\}$$

$$\sum_{i=1}^{n} x_{iu} = 1 \quad \forall u \in \{1, \ldots, n\}$$

$$x_{iu} \in \{0,1\} \quad \forall i, u \in \{1, \ldots, n\}^2$$

Subproblem

$$LAP_{GLB}^{iu} : l_{iu} = \min \sum_{j=1, j \neq i}^{n} \sum_{v=1, v \neq u}^{n} a_{ij} b_{uv} x_{jv}$$

$$\sum_{v=1, v \neq u}^{n} x_{jv} = 1 \quad \forall j \in \{1, \ldots, n\} \setminus \{i\}$$

$$\sum_{j=1, j \neq i}^{n} x_{jv} = 1 \quad \forall v \in \{1, \ldots, n\} \setminus \{u\}$$

$$x_{jv} \in \{0,1\} \quad \forall j \in \{1, \ldots, n\} \setminus \{i\}, v \in \{1, \ldots, n\} \setminus \{u\}$$

Polynomial complexity

Gilmore Lawler Bound (GLB) for QAP

(Gilmore, Journal of the society for industrial and applied mathematics, 1962)

Step 2: solves 1 linear assignment problem (LAP)

Original Problem

$$q = \min \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{u=1}^{n} \sum_{v=1}^{n} a_{ij} b_{uv} x_{iu} x_{jv} + \sum_{i=1}^{n} \sum_{u=1}^{n} c_{iu} x_{iu}$$

$$\sum_{u=1}^{n} x_{iu} = 1 \quad \forall i \in \{1, \ldots, n\}$$

$$\sum_{i=1}^{n} x_{iu} = 1 \quad \forall u \in \{1, \ldots, n\}$$

$$x_{iu} \in \{0, 1\} \quad \forall i, u \in \{1, \ldots, n\}^2$$

Relaxation Problem

$$glb = \min \sum_{i=1}^{n} \sum_{u=1}^{n} (l_{iu} + c_{iu}) x_{iu}$$

$$\sum_{u=1}^{n} x_{iu} = 1 \quad \forall i \in \{1, \ldots, n\}$$

$$\sum_{i=1}^{n} x_{iu} = 1 \quad \forall u \in \{1, \ldots, n\}$$

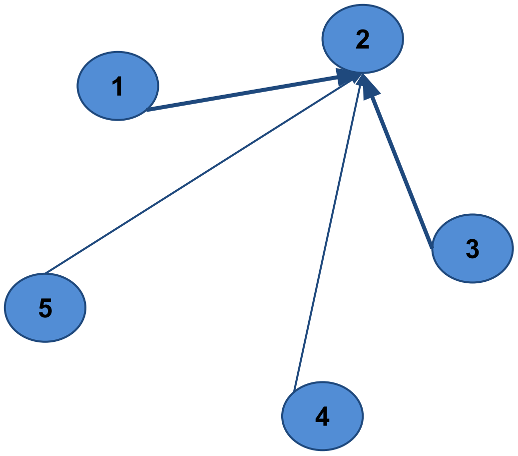
$$x_{iu} \in \{0, 1\} \quad \forall i, u \in \{1, \ldots, n\}^2$$

Polynomial complexity

Gilmore Lawler Bound (GLB) for QAP

Example1

Subproblem with i=2, u=4,
$x_{24}=1$



restricted flow matrix a_{2j}

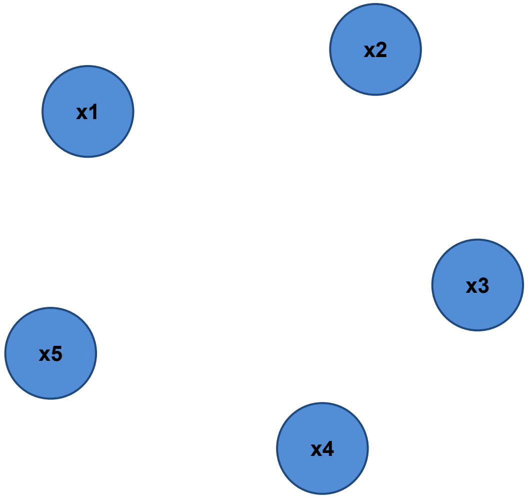
	0	0	0	0	0
i=2	2	0	2	1	1
	0	0	0	0	0
	0	0	0	0	0
	0	0	0	0	0

restricted distance matrix b_{4v}

	0	0	0	0	0
	0	0	0	0	0
	0	0	0	0	0
u=4	3	8	2	0	9
	0	0	0	0	0

Quadratic Assignment Problem as a Cost Function Network

Variables (facilities) $X_1,\ldots,X_n$ with domain values (locations) in 1..n



Cost Function Network

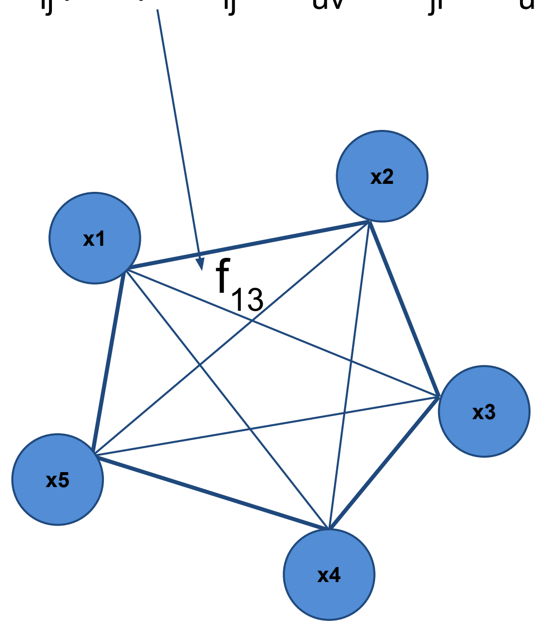
- X : set of variables
- D : set of finite domains
- F : set of cost functions
- $\top$: cost of forbidden tuples
- f_0 : global lower bound

Goal : min sum(F)

Quadratic Assignment Problem as a Cost Function Network

Variables (facilities) $X_1,\dots,X_n$ with domain values (locations) in 1..n

Binary cost functions $f_{ij}(u,v) = a_{ij} * b_{uv} + a_{ji} * b_{uv}$ with $i<j$ (non oriented model)



Cost Function Network

- X : set of variables
- D : set of finite domains
- F : set of cost functions
- $\top$: cost of forbidden tuples
- f_0 : global lower bound

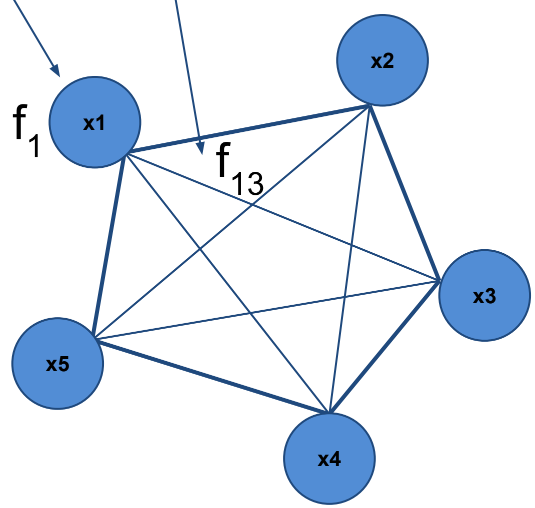
Goal : min sum(F)

Quadratic Assignment Problem as a Cost Function Network

Variables (facilities) $X_1,\dots,X_n$ with domain values (locations) in 1..n

Binary cost functions $f_{ij}(u,v)= a_{ij} * b_{uv} + a_{ji} * b_{uv}$ with i<j (non oriented model)

Unary cost functions $f_i(u) = c_{iu}$



Cost Function Network

- X : set of variables
- D : set of finite domains
- F : set of cost functions
- $\top$: cost of forbidden tuples
- f_0 : global lower bound

Goal : min sum(F)

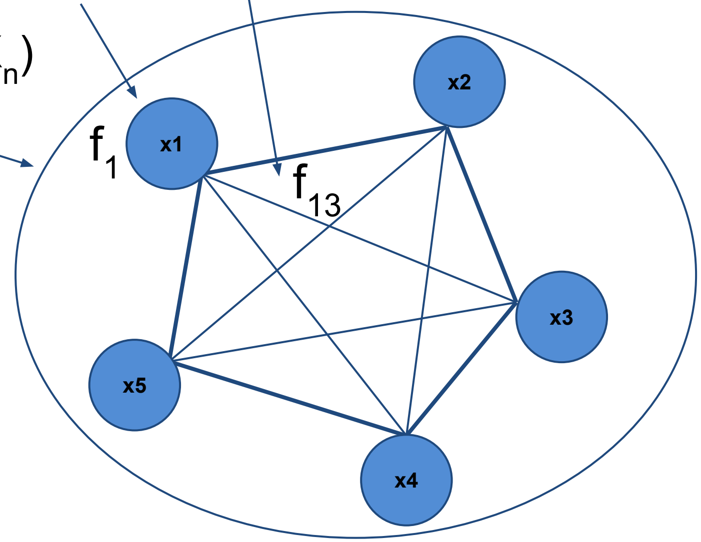
Quadratic Assignment Problem as a Cost Function Network

Variables (facilities) $X_1,...,X_n$ with domain values (locations) in 1..n

Binary cost functions $f_{ij}(u,v)= a_{ij} * b_{uv} + a_{ji} * b_{uv}$ with $i<j$ (non oriented model)

Unary cost functions $f_i(u) = c_{iu}$

AllDifferent($X_1,...,X_n$)



Cost Function Network

- X : set of variables
- D : set of finite domains
- F : set of cost functions
- $\top$: cost of forbidden tuples
- f_0 : global lower bound

Goal : min sum(F)

*can be generalized to an arbitrary 4-dimension tensor q_{ijuv} replacing a_{ij}*b_{uv}*

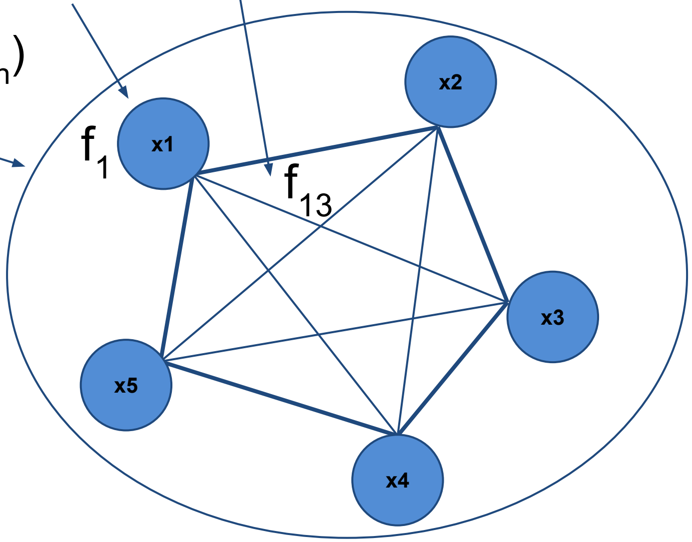
Quadratic Assignment Problem as a Cost Function Network

Variables (facilities) $X_1,\dots,X_n$ with domain values (locations) in 1..n

Binary cost functions $f_{ij}(u,v) = a_{ij} * b_{uv} + a_{ji} * b_{uv}$ with i<j (non oriented model)

Unary cost functions $f_i(u) = c_{iu}$

AllDifferent($X_1,\dots,X_n$)



Cost Function Network

- X : set of variables
- D : set of finite domains
- F : set of cost functions
- $\top$: cost of forbidden tuples
- f_0 : global lower bound

Goal : min sum(F)

Lower bounds obtained by problem reformulations in polynomial time

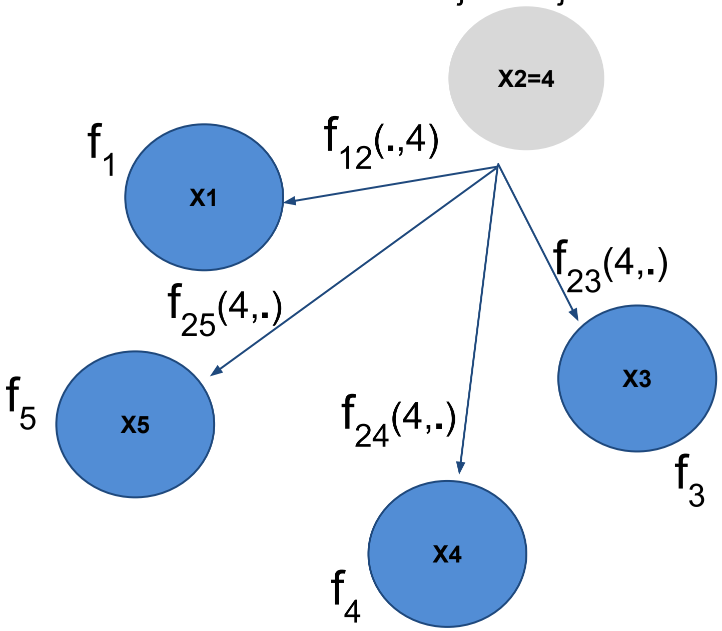
- *Table cost functions (Cooper et al, AIJ 2010)*
- AllDifferent($X_1,\dots,X_n$) + $\sum_{i=1}^{n} f_i$ *(Sewa et al, AAAI 2026)*

*can be generalized to an arbitrary 4-dimension tensor q_{ijuv} replacing a_{ij}*b_{uv}*

Singleton Node Consistency (SNC-Greedy)

Subproblem with X_2=4

Projects f_{2j} on f_j for every $j \neq i$



Projection

We move cost from the higher-arity function to the lower-arity function.

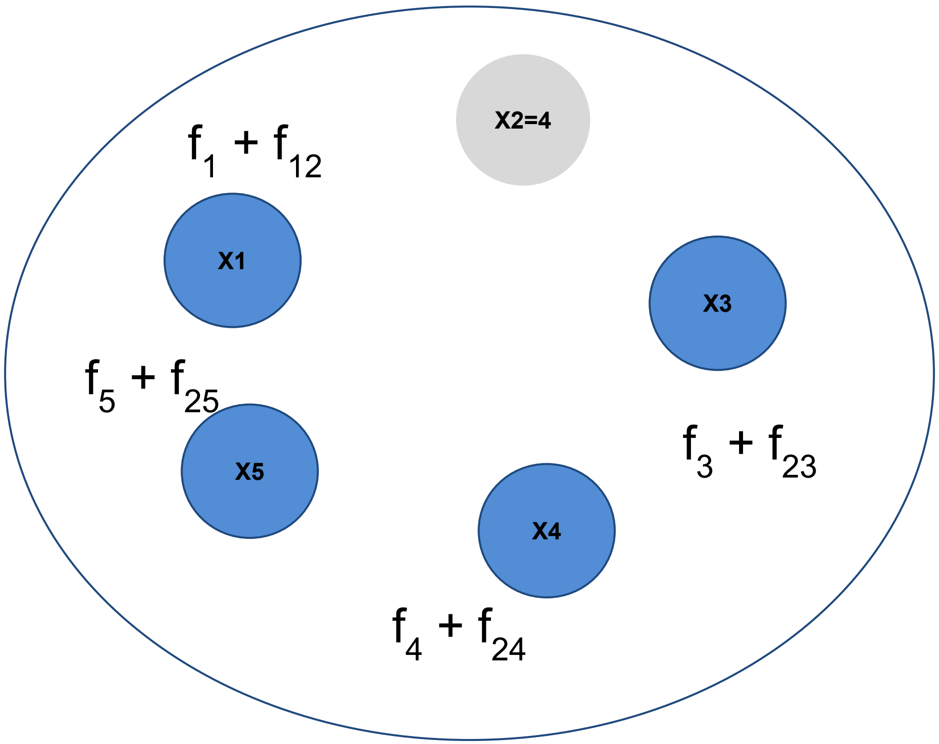
Extension

We move cost from the lower-arity function to the higher-arity function

Singleton Node Consistency (SNC-Greedy)

Subproblem with X_2=4

Reformulates using AllDifferent *(Sewa et al, AAAI 2026)*



Reformulation

- Solve the Linear Assignment Problem using the unary cost function as the cost matrix :

$$z^* \leftarrow LAP(f_i)$$

- Update the unary cost functions using reduced costs :

$$f_i(j) \leftarrow f_i(j) - u_i^* - v_j^*$$

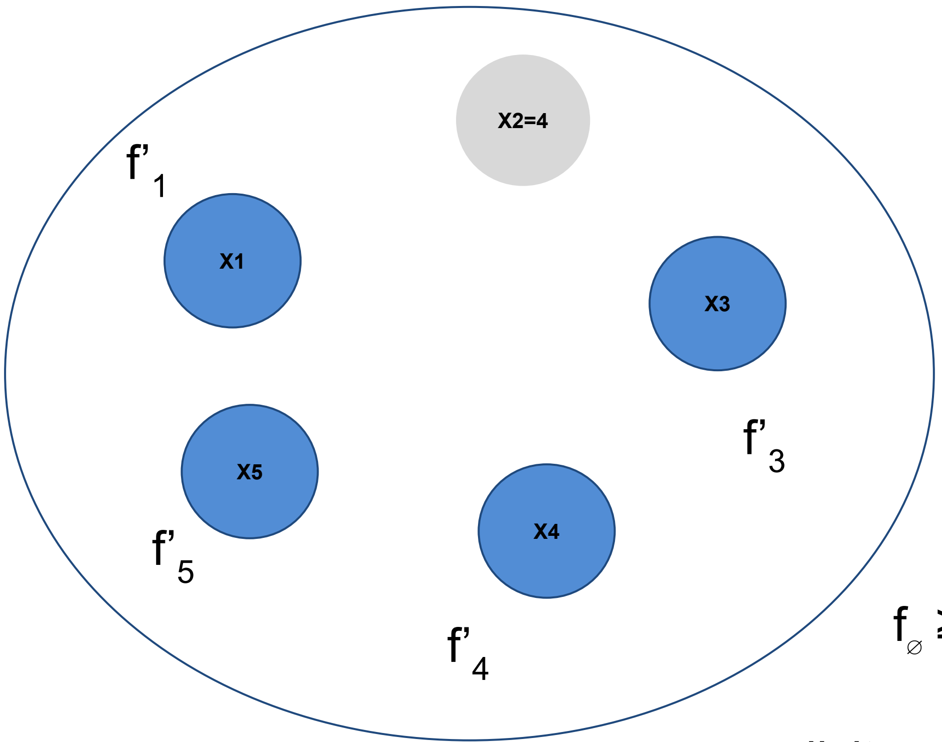
- Increase the global lower bound :

$$f_\emptyset \leftarrow f_\emptyset + z^*$$

Singleton Node Consistency (SNC-Greedy)

Subproblem with X_2=4

Reformulates using AllDifferent *(Sewa et al, AAAI 2026)*



$f_\varnothing \geq f_2(4)$

explicit problem lower bound

Reformulation

- Solve the Linear Assignment Problem using the unary cost function as the cost matrix :

$$z^* \leftarrow LAP(f_i)$$

- Update the unary cost functions using reduced costs :

$$f_i(j) \leftarrow f_i(j) - u_i^* - v_j^*$$

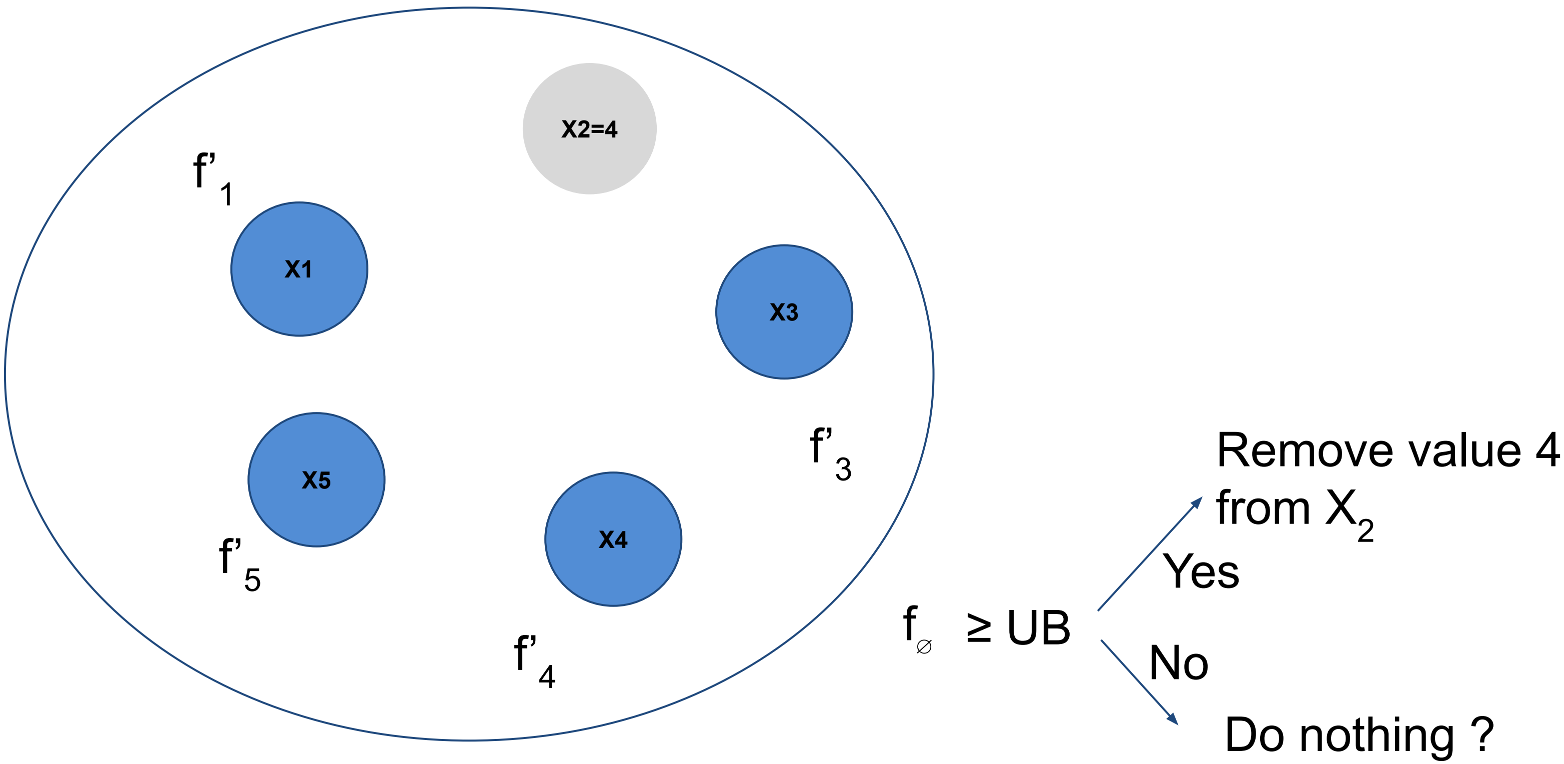
- Increase the global lower bound :

$$f_\emptyset \leftarrow f_\emptyset + z^*$$

Singleton Node Consistency (SNC-Greedy)

Subproblem with X_2=4

Reformulates using AllDifferent *(Sewa et al, AAAI 2026)*

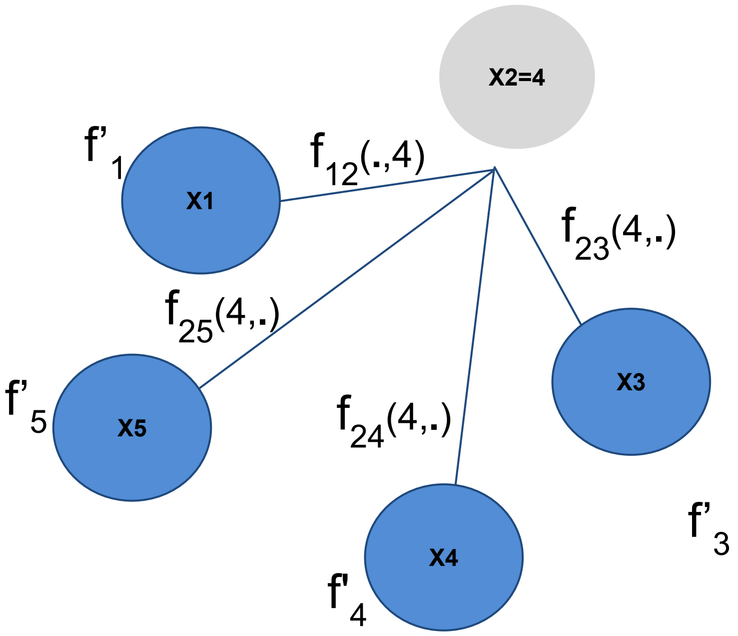


Singleton Consistency in CSP only used for pruning (Debruyne, 1997)

Singleton Node Consistency (SNC-Greedy)

Transfers the reformulated subproblem X_2=4 to the original problem

Extends from f'_j to f_{2j} for every $j \neq i$

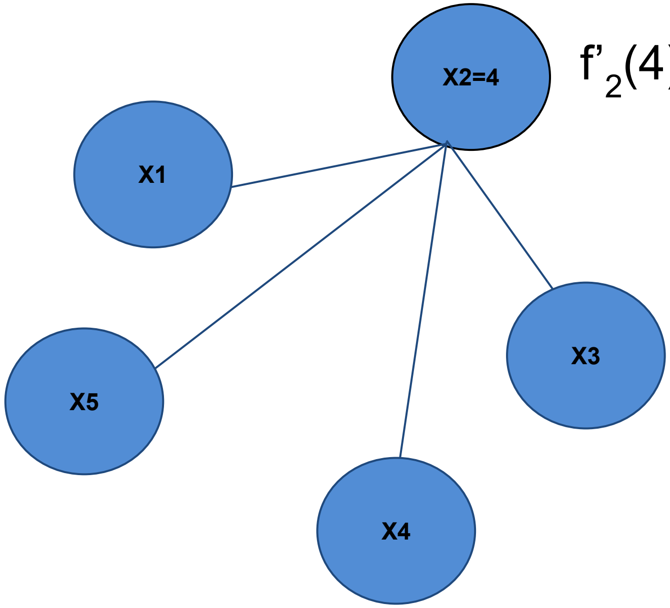


Singleton Node Consistency (SNC-Greedy)

Transfers the reformulated subproblem X_2=4 to the original problem

Extends from f'_j to f_{2j} for every $j \neq i$ and $f_\varnothing$ to f_2

$f'_2(4) \leftarrow f_\varnothing + f_2(4)$



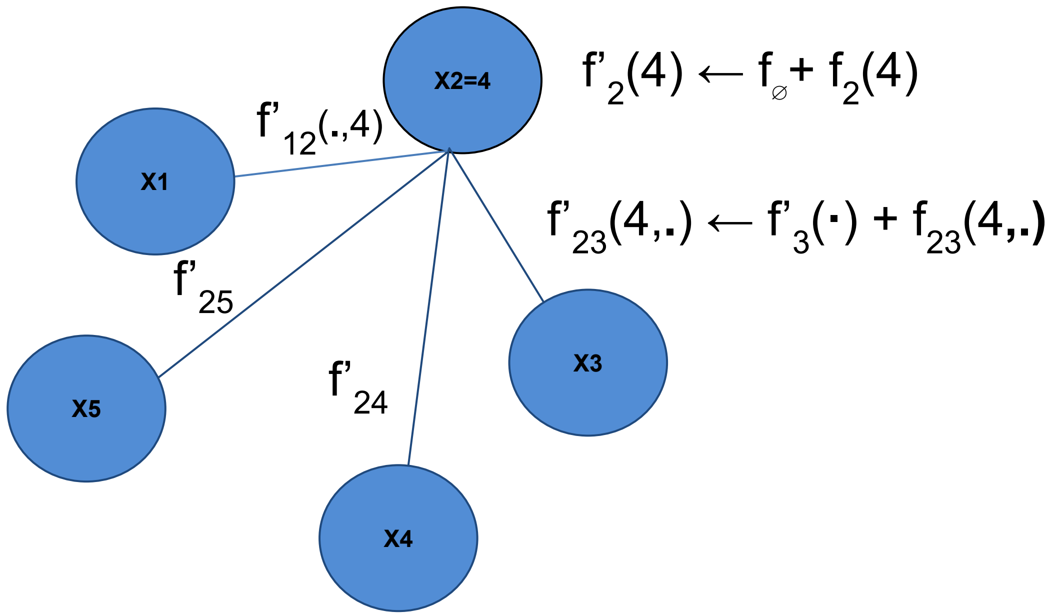
SNC can increase the problem lower bound if :

- all values have their unary cost increased

Singleton Node Consistency (SNC-Greedy)

Transfers the reformulated subproblem X_2=4 to the original problem

Extends from f'_j to f_{2j} for every $j \neq i$ and $f_\varnothing$ to f_2



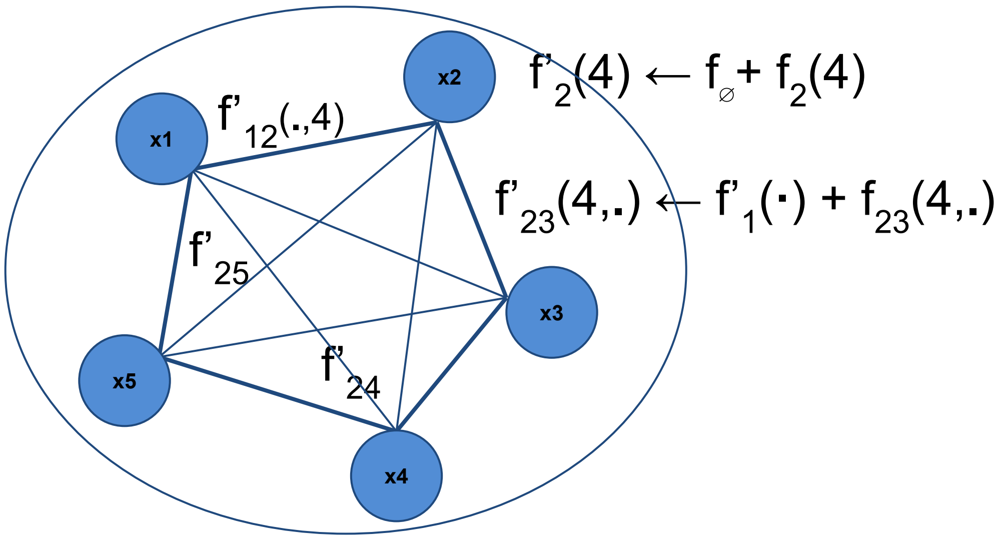
SNC can increase the problem lower bound if :

- all values have their unary cost increased
- binary cost functions find a better reformulation

Singleton Node Consistency (SNC-Greedy)

Transfers the reformulated subproblem X_2=4 to the original problem

Extends from f'_j to f_{2j} for every $j \neq i$ and $f_\varnothing$ to f_2



SNC can increase the problem lower bound if :

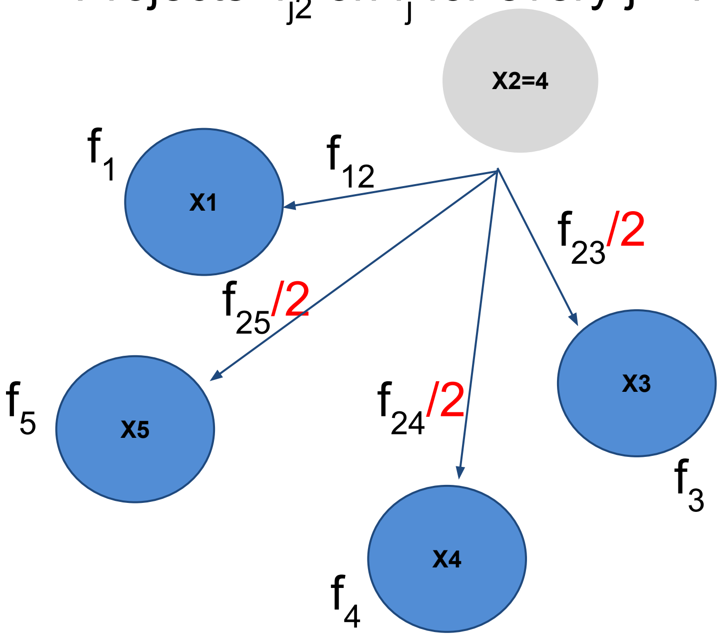
- all values have their unary cost increased
- binary cost functions find a better reformulation
- AllDifferent find a better reformulation

Repeat until a fixpoint is reached

Singleton Node Consistency (SNC-GLB)

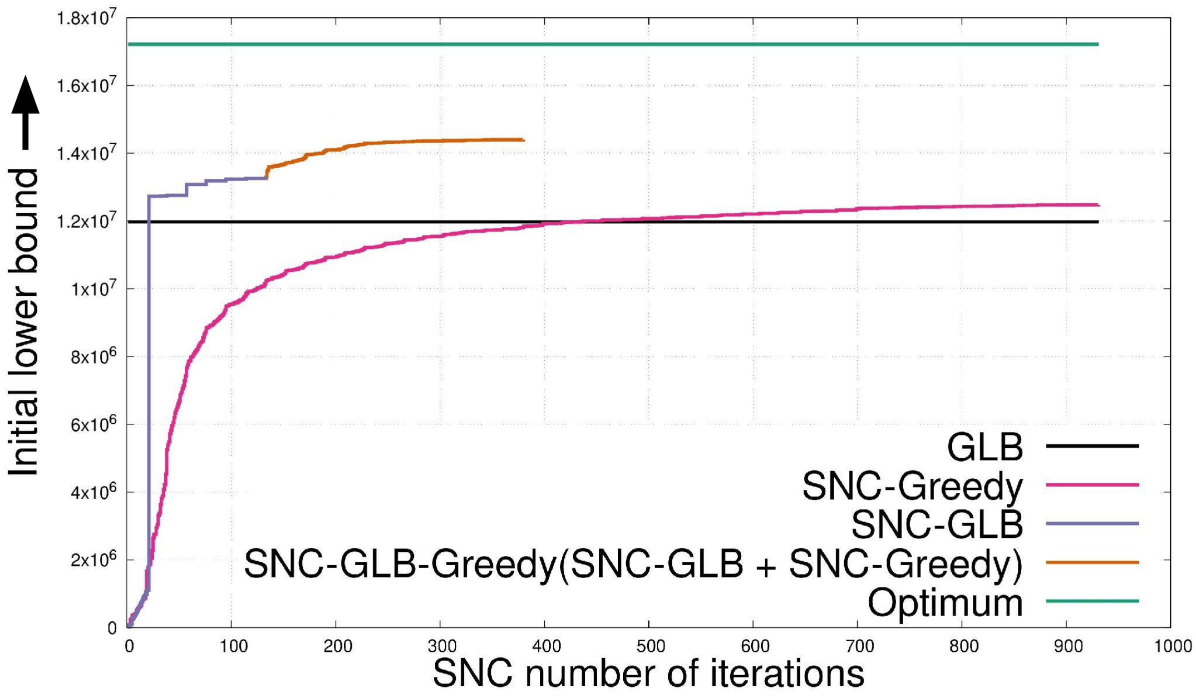
Subproblem with X_2=4

Projects f_{j2} on f_j for every j < i and half of f_{2j} on f_j for every j > i



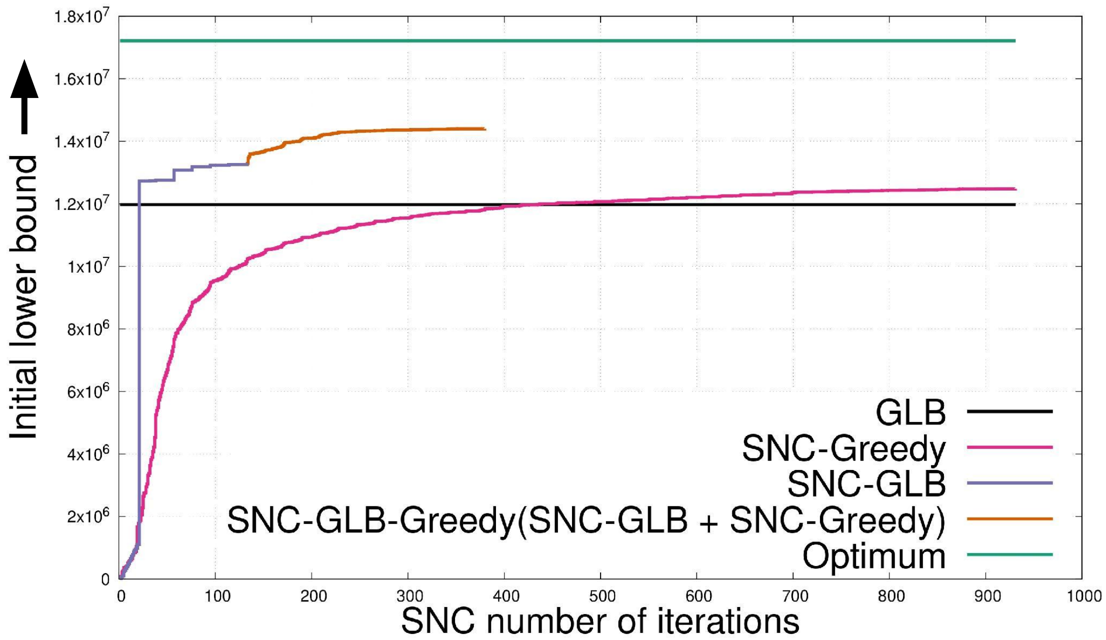
Theorem: SNC-GLB will find a problem lower bound equal or greater than GLB
(hypothesis: symmetric distance and flow matrices)

Experimental Results on QAPLIB instance els19



SNC-GLB-Greedy took 0.69s and 380 iter. in preprocessing and then TB2 solved in 5.82s.

Experimental Results on QAPLIB instance els19



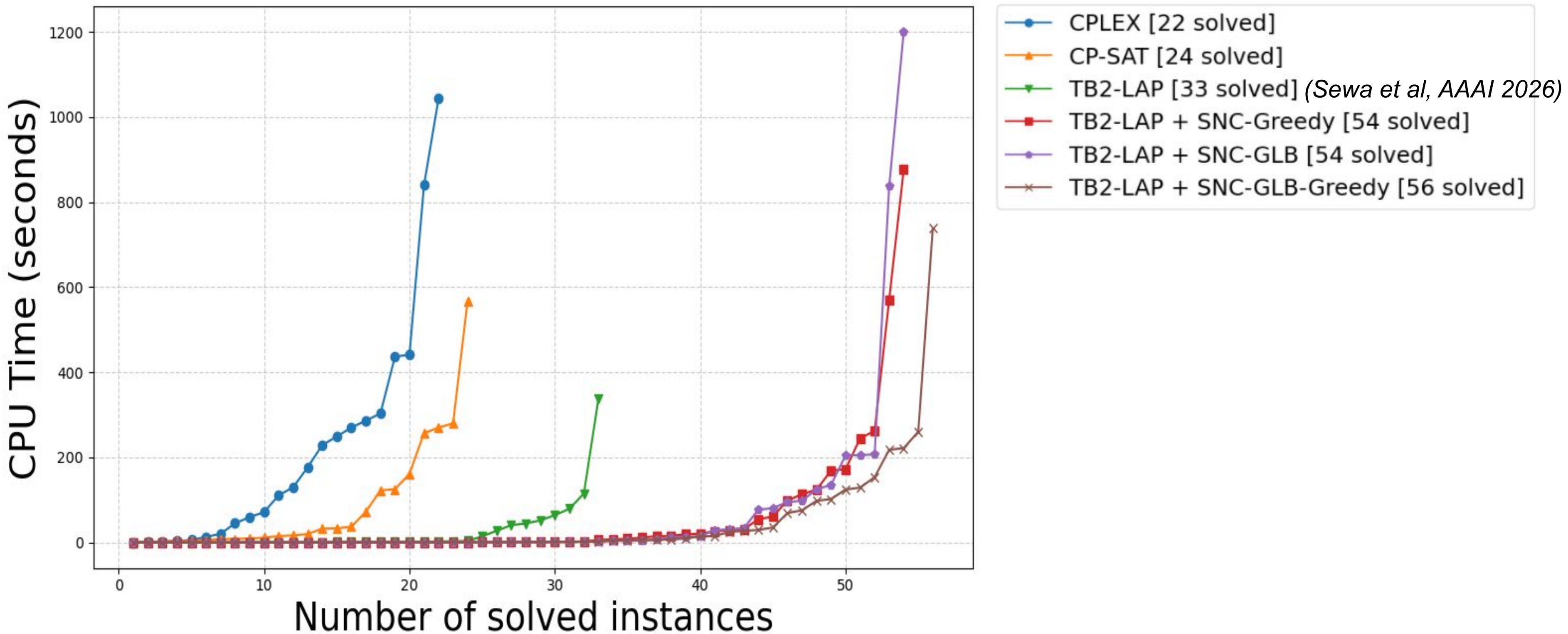
SNC-GLB-Greedy took 0.69s and 380 iter. in preprocessing and then TB2 solved in 5.82s.

Average initial lower bound gap ($1 - \frac{LB}{bestsolution}$) on 132 instances

Methods (Preprocessing)	**Average GAP (%)** ↘
GLB	30.46
SNC-Greedy	26.38
SNC-GLB	26.32
SNC-GLB-Greedy	**24.13**

Experimental Results on QAPLIB (Cactus plot)

Total Instances : 132 (≤ 100 variables)
Time limit : 1200 seconds
Largest instance solved : 90 variables (lipa90)



Original model with AllDifferent

Quadratic Assignment Problem as a Cost Function Network

Inverted compact model

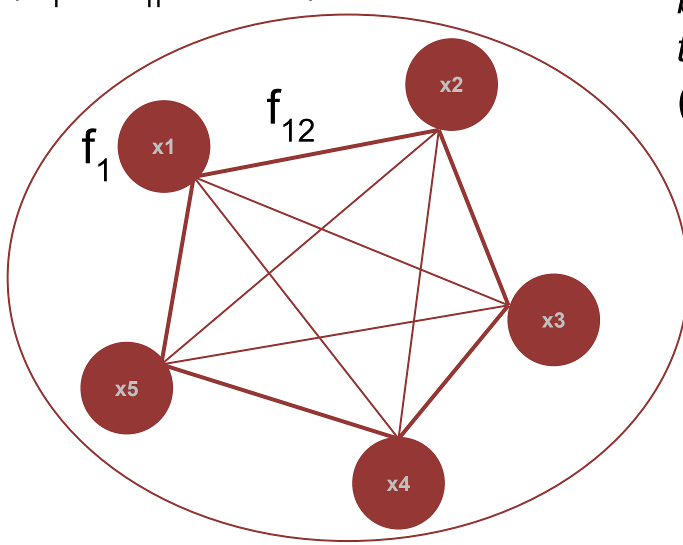
Variables (**locations**) $X_1,...,X_n$ with domain values (**facility classes**) in 1..bound

Binary cost functions $f_{uv}(i,j)= a_{ij} * b_{uv} + a_{ij} * b_{vu}$ with u<v (non oriented model)

Unary cost functions $f_u(i) = c_{iu}$

GlobalCardinalityConstraint($X_1,...,X_n$,bounds)

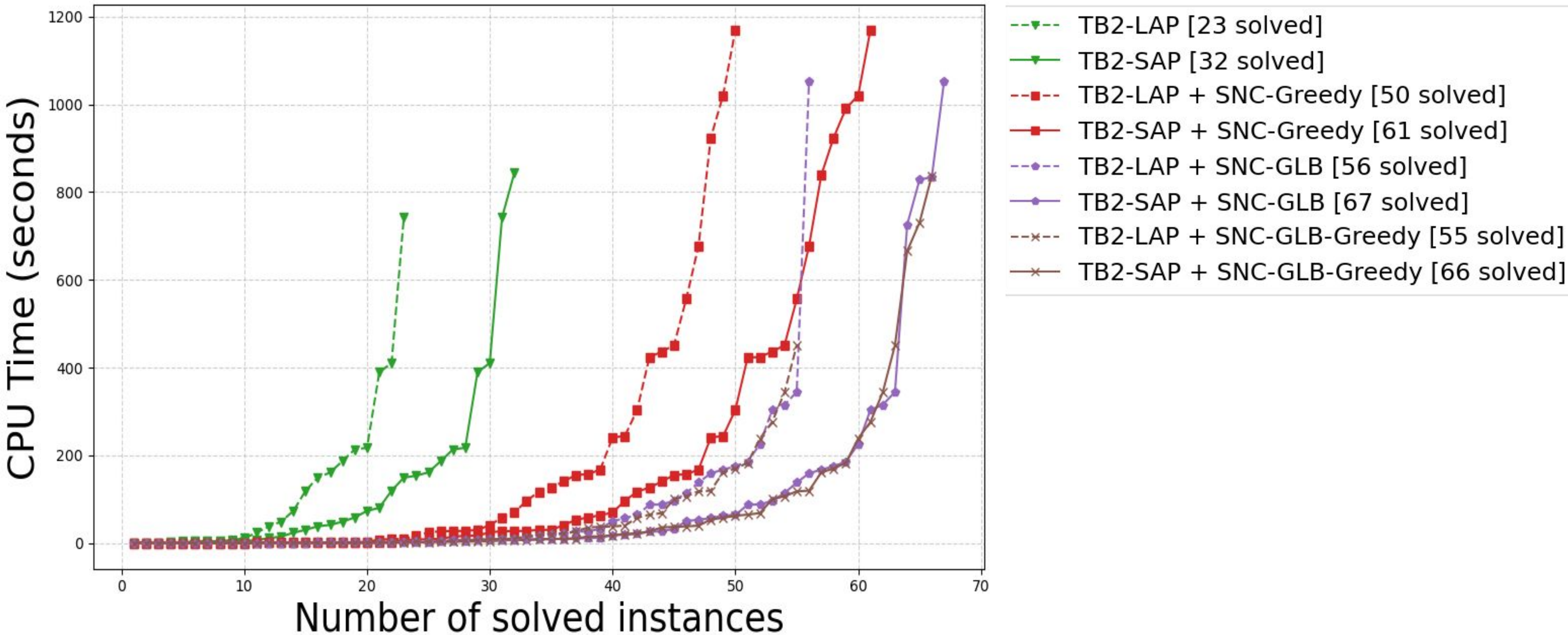
by exploiting symmetries in the distance and flow matrix (Fischetti et al, 2012)



SNC-Greedy and SNC-GLB can be applied to the inverted compact model
(Semi-Assignment Problem solved in polynomial time)

Experimental Results on QAPLIB (Cactus plot)

Total Instances : 132 (≤ 100 variables)
Time limit : 1200 seconds
Largest instances solved : 90 variables (lipa90)



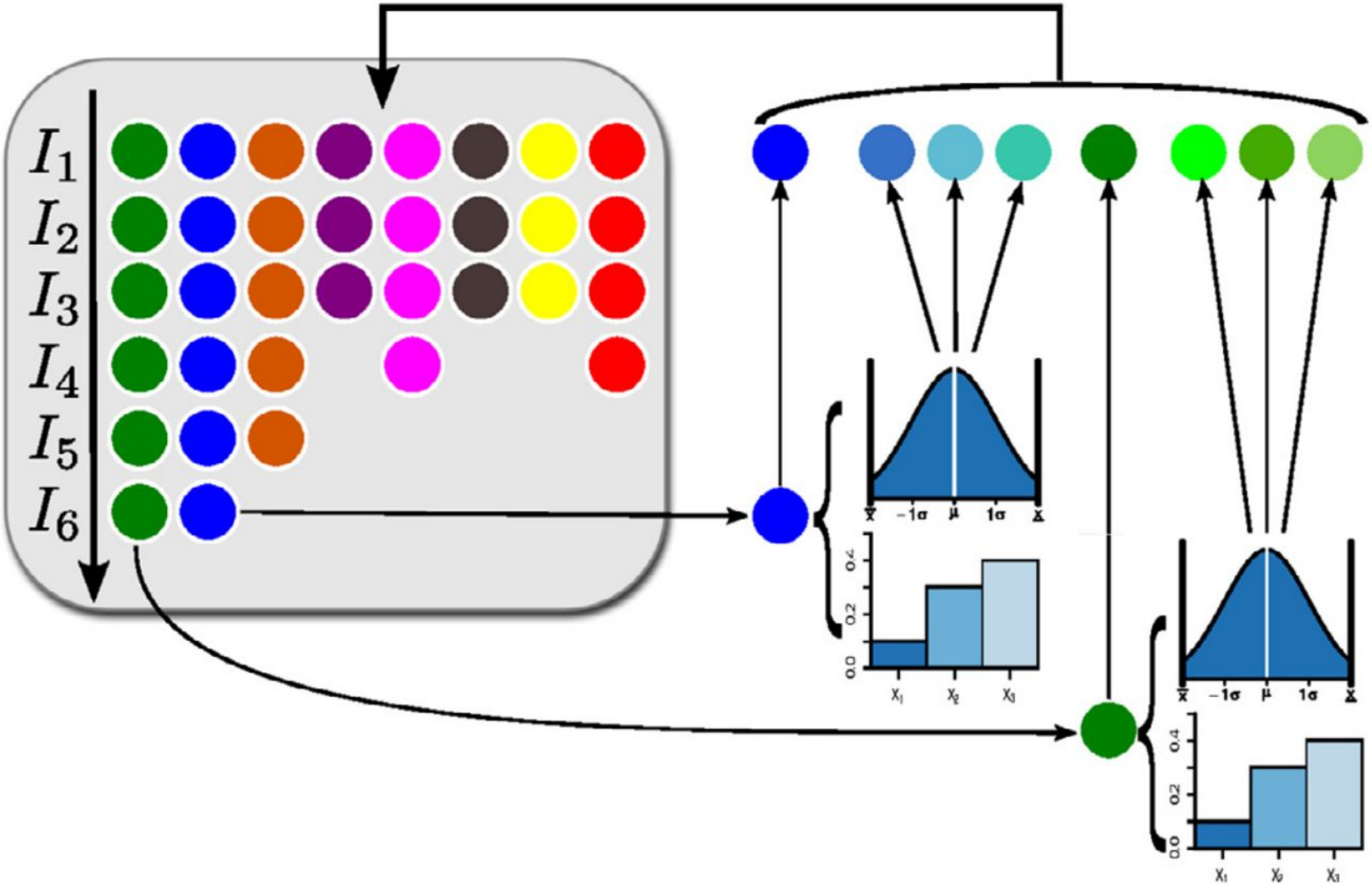
Inverted model with AllDifferent (LAP) and GCC (SAP)

Comparison of number of instances best solved (in terms of solution quality and then solving time) on four different problem modelling

QAPLIB Instances	P_{alldiff}	$P^{\tau}_{\text{alldiff}}$	P_{knapsack}	P_{binary}
bur26*		**8**		
chr*	**14**			
els19		**1**		
esc*	**8**	4	1	5
had*	1	**3**		
kra*	**2**	1		
lipa*	1	**15**		
nug*	6	**9**		
rou*	**3**			
scr*	**3**			
sko*	**12**	1		
ste36*		**3**		
tai*	**14**	11		1
tho*		**2**		
wil*	1	1		

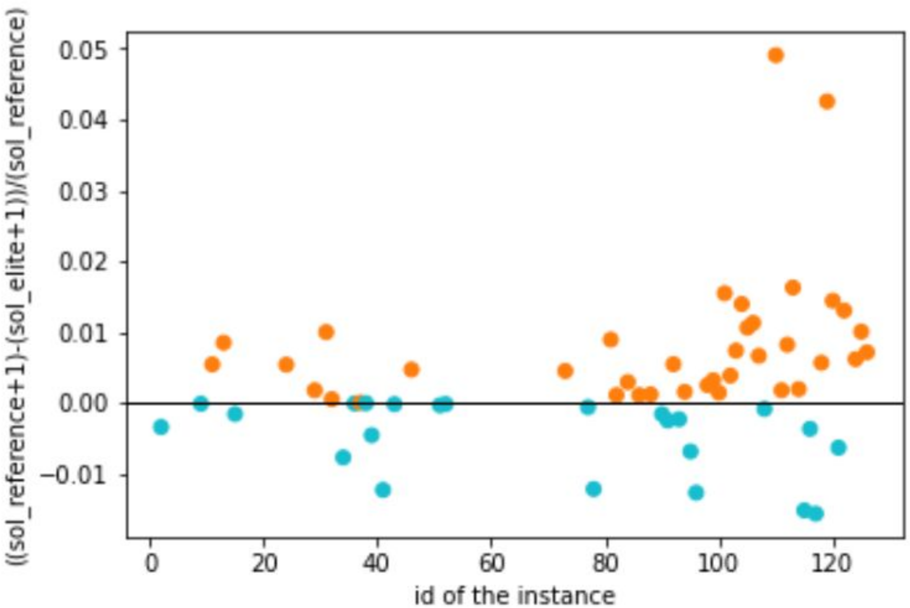
irace: Iterated racing for automatic algorithm configuration

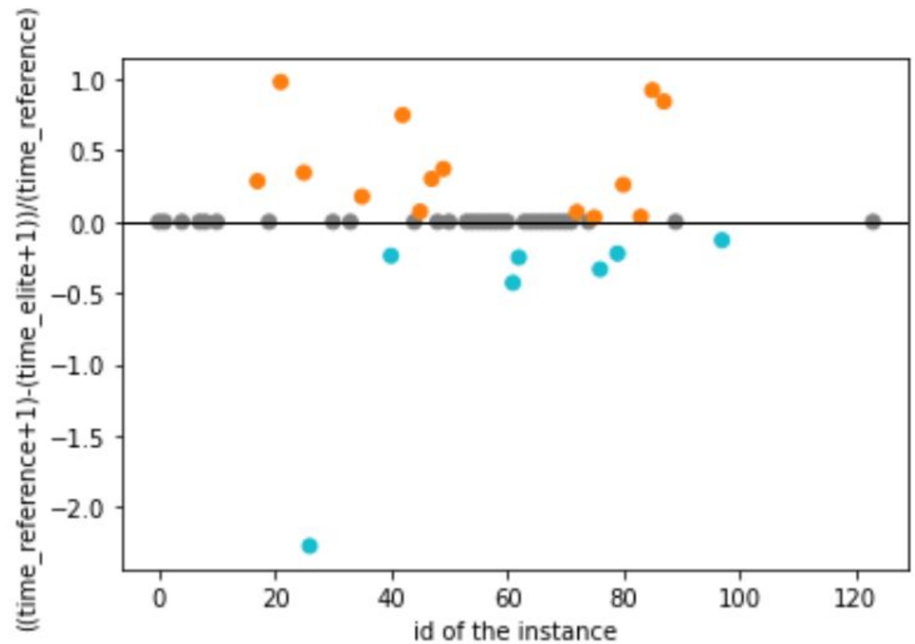
(López-Ibáñez et al, 2016)



irace execution trace

Tuning six categorical parameters of toulbar2 (variable and value search heuristics, DAC ordering, SNC preprocessing) (768 combinations)



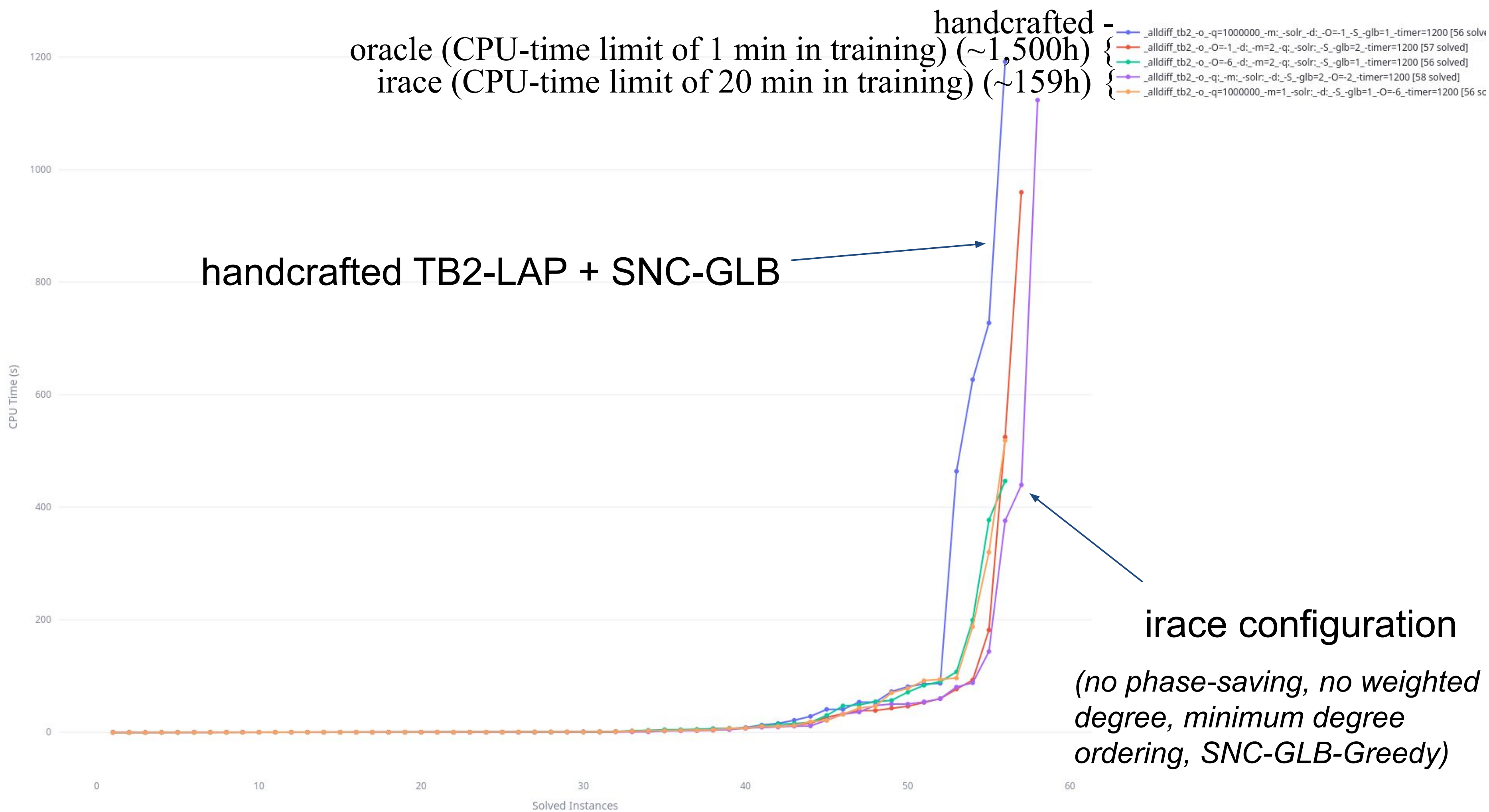


Relative objective value gap to handcrafted TB2-LAP + SNC-GLB

Relative time gap to handcrafted TB2-LAP + SNC-GLB

orange: irace better than baseline, blue: baseline better

Cactus plot of handcrafted/oracle/irace configurations



Optimality gap percentage
and number of solved/feasible instances

Gap	mean	median	std	Score	Nb. instances
CPLEX	16.41	9.35	20.38	25.00	109
CP-SAT	5.12	2.29	11.00	33.00	122
LAP	4.01	1.88	5.27	41.50	**132**
SNC-LAP-Greedy	1.44	**0.00**	**2.29**	83.00	131
SNC-LAP-GLB	1.50	**0.00**	2.56	89.00	**132**
SNC-LAP-GLB-Greedy	**1.42**	**0.00**	2.33	**92.00**	131

TB2 handcrafted

Gap	mean	std	Nb. solved instances	Nb. feasible instances
θ^{def}(handcrafted)	1.56	2.61	56	**132**
θ_1^{irace}	2.04	8.79	**58**	131
θ_2^{irace}	**1.31**	**2.02**	56	**132**
$\theta_1^{oracle1min}$	2.00	8.78	57	131
$\theta_2^{oracle1min}$	1.34	2.22	56	**132**
Hexaly	**1.09**	3.53	01	**132**

TB2

Time limit : 1200 seconds

Conclusion and perspectives

Conclusion

- Hybridizing **CP** (SNC) and **IP** (GLB) techniques significantly improved the **CFN solver** for the generalized QAP.
- Generalized preprocessing algorithm (SNC) to handle problems involving global constraints(Alldifferent, GCC …)
- Improved results on the instances (**bur26***, **lipa***, and **tai64c**) compared to specialized QAP solvers (Fischetti et al. (2012) and Erdogan et al. (2007)).
- irace found better parameter configurations than those defined manually.

Perspectives

- Explore other methods and problem formulation to get stronger bounds.
- Conflict explanations using weighted degree heuristic.
- Dynamic parameter configurations using portfolios.

References

Koopmans, T.C., Beckmann, M.: Assignment Problems and the location of eco-nomic activities. Econometrica: journal of the Econometric Society pp. 53–76(1957)

Gilmore, Optimal and suboptimal algorithms for the Quadratic Assignment Problem. Journal of the society for industrial and applied mathematics (1962)

Focacci, F, Andrea Lodi, and Michela Milano. *Cost-based domain filtering.* In CP’99, pages 189–203. Springer, 1999.

Burkard, Dell’Amico, Martello. Assignment problems: revised reprint. SIAM (2012).

Cooper, de Givry, Sanchez, Schiex, Zytnicki, Werner. Soft arc consistency revisited. Artificial Intelligence Journal (2010)

Jonker, Volgenant. A shortest augmenting path algorithm for dense and sparse linear assignment problems. Computing (1987).

López-Ibáñez, Dubois-Lacoste, Pérez Cáceres, Birattari, Stützle. The irace package: Iterated racing for automatic algorithm configuration. Operations Research Perspectives (2016).

Sewa, Allouche, de Givry, Katsirelos, Montalbano, Schiex. Assignment problems in cost function networks. In: Proc. of AAAI-26. Singapore (2026).

Thank you for your attention !

guidio.sewa@inrae.fr

toulbar2: cost function network open source C++ solver

https://github.com/toulbar2/toulbar2

Git: git clone https://github.com/toulbar2/toulbar2.git

wcsp instances:
https://forge.inrae.fr/thomas.schiex/cost-function-library/-/tree/master/qaplib/instances/wcsp

Install on linux: sudo apt install toulbar2
Python interface: pip install pytoular2

Many problem input formats available:
cfn, cnf, lp, opb, qpbo, uai, wbo, wcnf, wcsp, xcsp3