Metaheuristics for scheduling production in large-scale open-pit mines accounting for metal uncertainty - Tabu search as an example

Amina Lamghari

McGill

COSMO – *Stochastic Mine Planning Laboratory*

Department of Mining and Materials Engineering

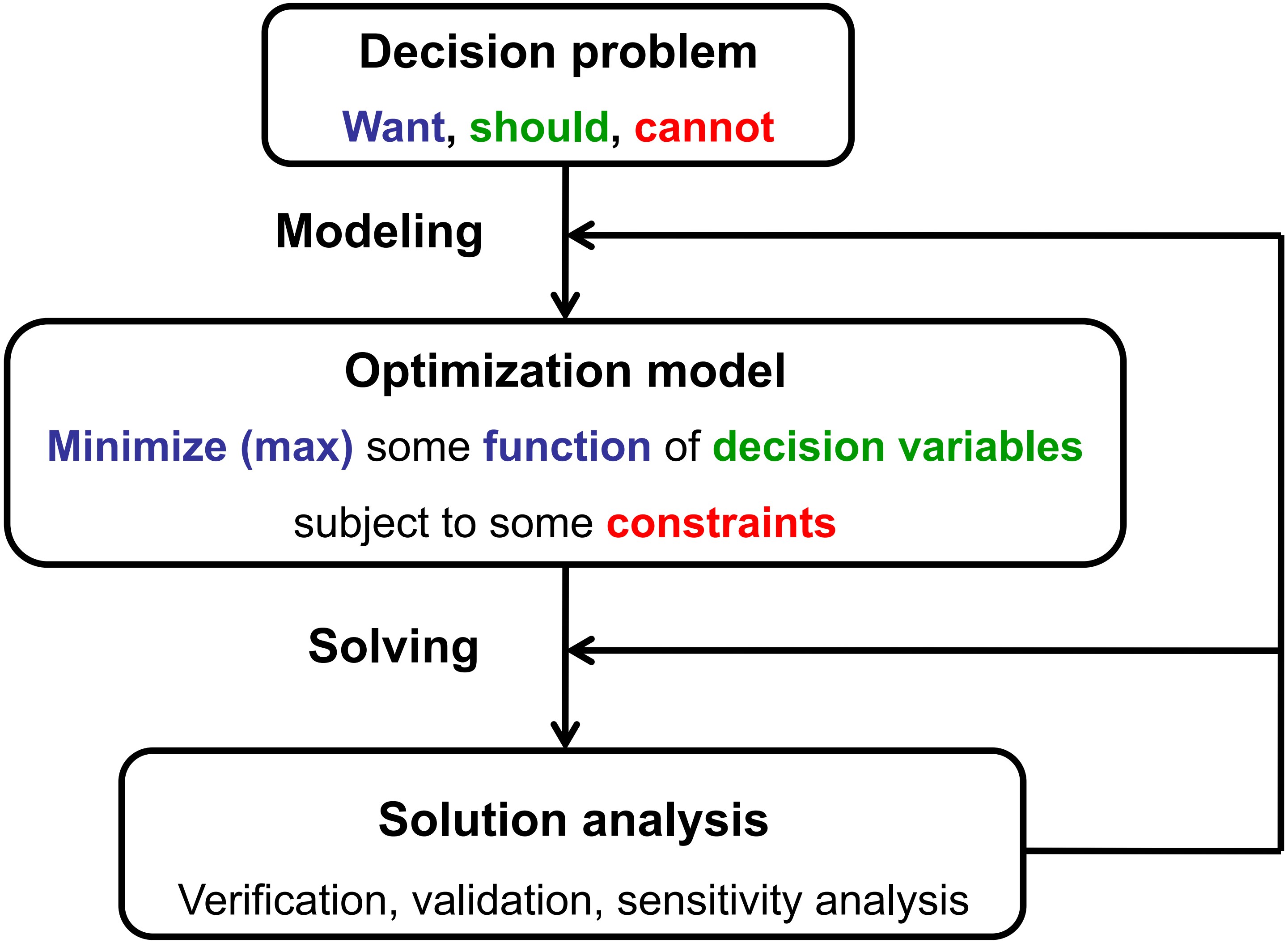
Outline

- Introduction
- Overview of Tabu Search
- Adaptation of Tabu Search to solve the open-pit mine production scheduling problem with metal uncertainty
- Conclusions

Not an exhaustive presentation

Only an introduction to some basic ideas and principles

The big picture



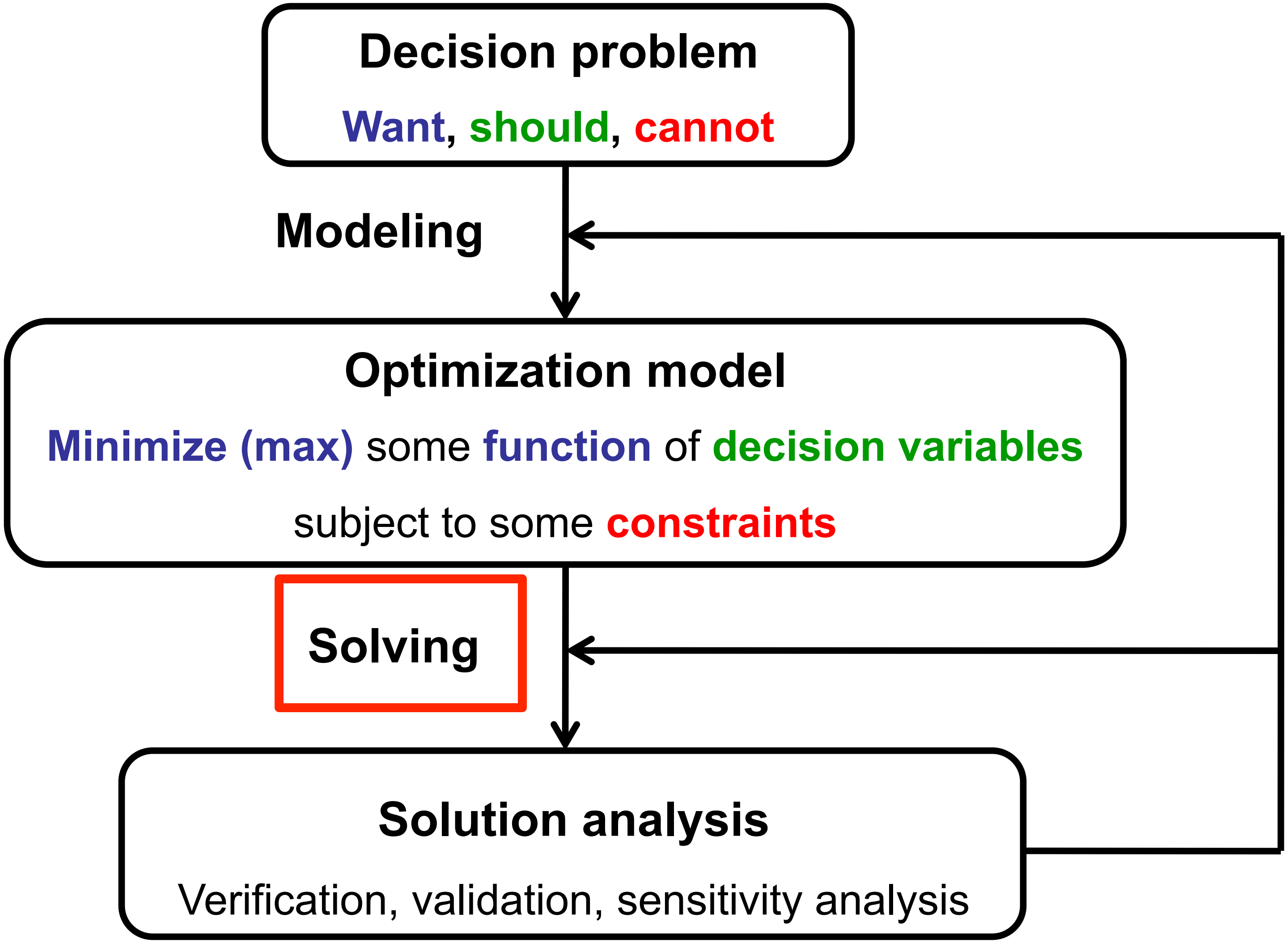
Optimization models

Minimize (maximize) some **function** of **decision variables** subject to some **constraints**

Continuous	**Discrete** (Integer, binary)
Single criterion	**Multi-criteria** (conflicting objectives)
Unconstrained or few constraints	**Many constraints** (difficult to find a feasible solution)
Linear	**Nonlinear**
Deterministic data	**Stochastic data**

Its probability distribution is known

The big picture

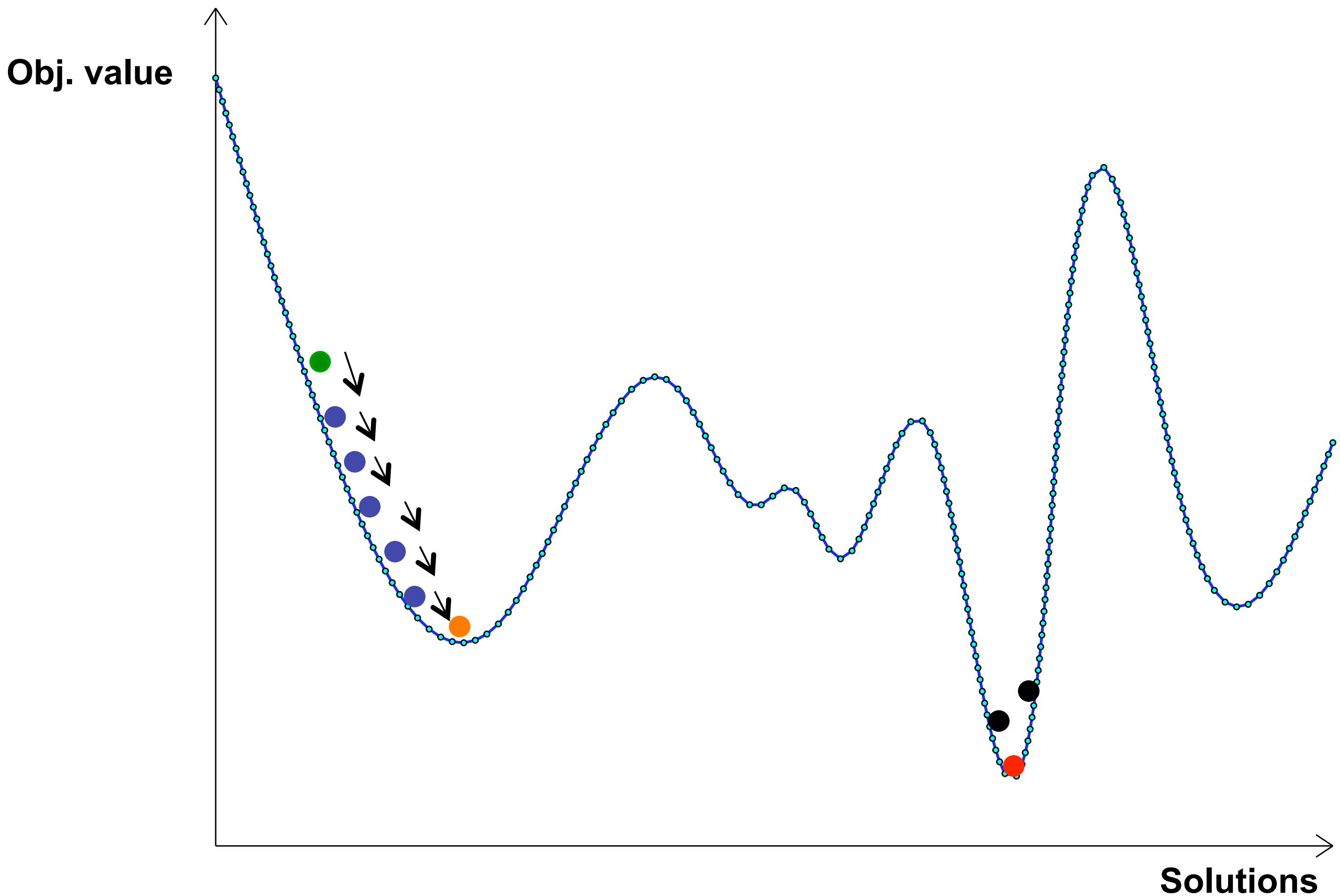


Solving optimization problems

Easy	Difficult
Continuous	Discrete (Integer, binary)
Single criterion	Multi-criteria (conflicting objectives)
Unconstrained or few constraints	Many constraints
Linear	Nonlinear
Deterministic data	Stochastic data
Exact solution procedures (Simplex, Branch & Bound…) **Not efficient for difficult large-scale problems** (prohibitive CPU time, memory…)	**Approximate solution techniques** (Heuristics / metaheuristics…) A heuristic: fits a specific problem and take advantage of its structure A metaheuristic: provides general structure and strategy guidelines **Used to rapidly come to a solution hoped to be close to the optimal**

The descent method

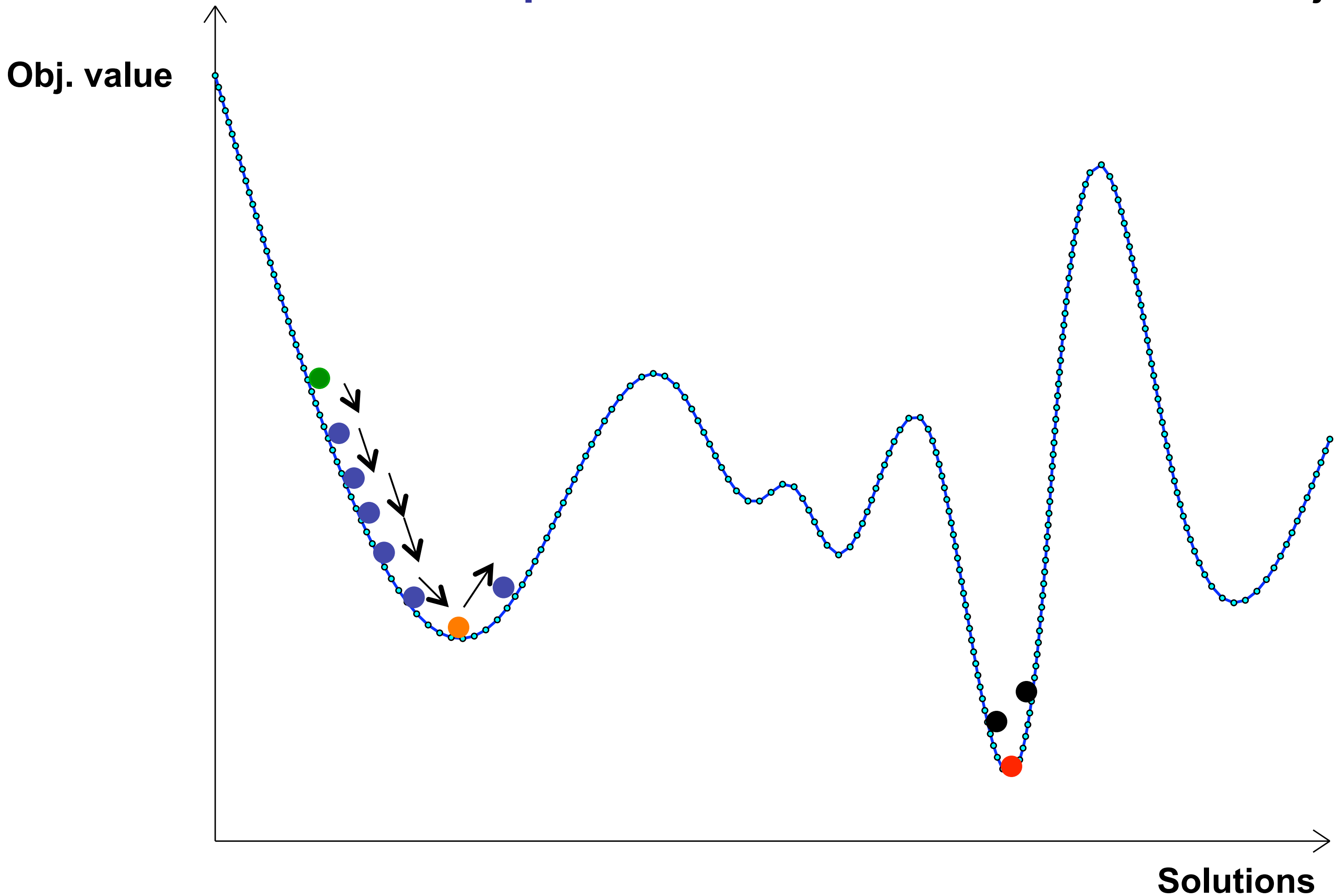
At each iteration,
Select the **best improving solution** x' in N(x) as the new current solution



The method stops at the **first local optimum** reached from the **starting solution**

Tabu search

To **avoid being trapped in a local optimum**, **move** to a new solution x' in N(x) **even if** this leads to **no improvement** or a **deterioration** of the objective function

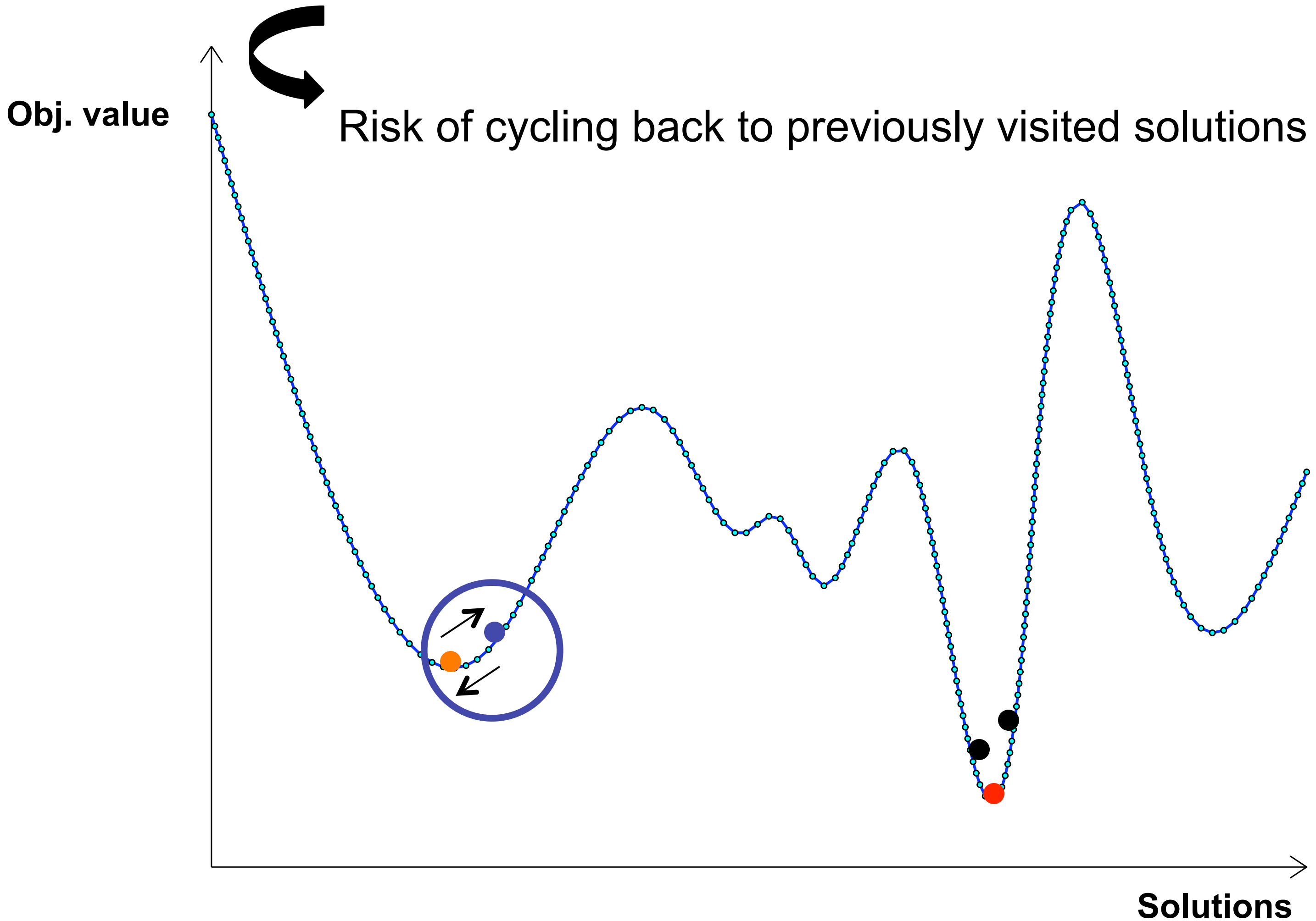


At each iteration, select the **best ~~improving~~ solution** x' in N(x)

The idea is nice but …

At each iteration, select the **best solution** x' in N(x)

Risk of cycling back to previously visited solutions



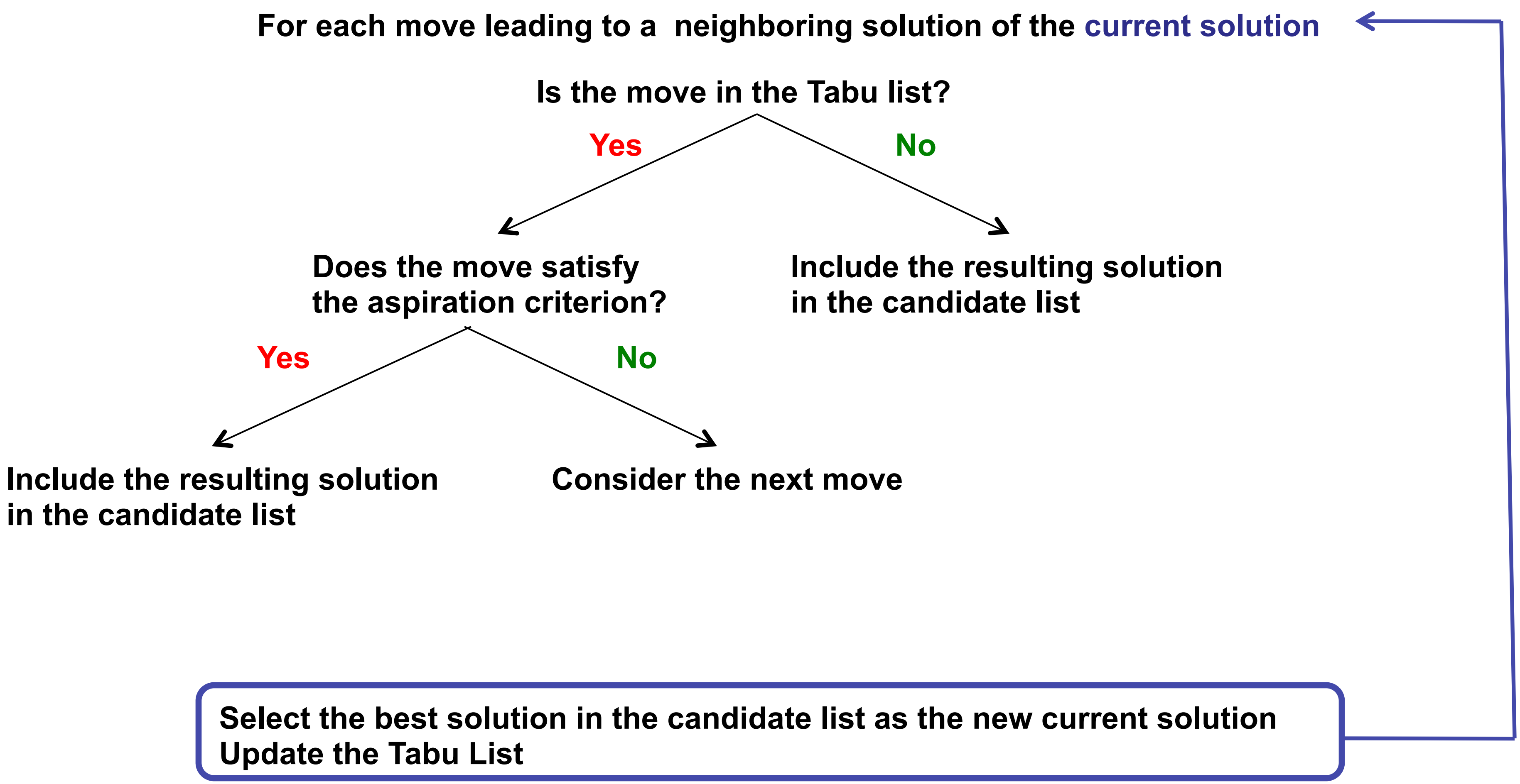
Avoiding cycles

- Use a list to keep track of the T recently visited solutions (history of the search)
- When considering a new solution, first check if it is in this list. If so, discard it
- Pro: instantly eliminates any possibility that a cycle of length T occurs
- Con: requires a lot of storage
- Con: checking whether a potential solution is in the list or not may be computationally costly

Avoiding cycles

- Record the recent moves performed on the current solution and forbid moves that reverse their effect
- These “forbidden” moves are declared **Tabu** and are disregarded when exploring the neihborhood for some number of iterations, called the **Tabu tenure** of the move
- Tabus are stored in a **short-term memory** of the search, called **Tabu list**
- Should be long enough to prevent cycling and short enough to not exclude more moves than necessary
- An **aspiration criterion** is used to override the Tabu status of some attractive solutions during the process

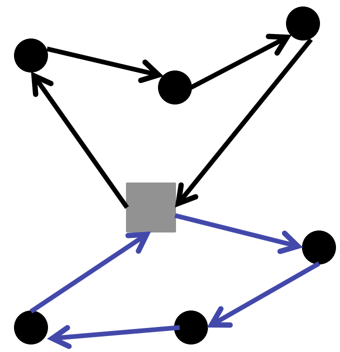
Exploring the neighborhood: decision tree



Stopping criteria

- A fixed number of iterations
- A fixed amount of CPU time
- After some number of iterations without an improvement in the objective function value
- When the objective reaches a pre-specified threshold value

An illustrative problem



The **Classical** Vehicle Routing Problem

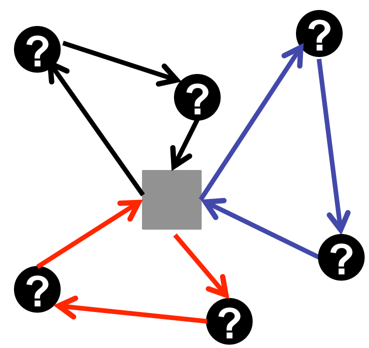
A depot at which a vehicle of limited capacity Q is based

A set of customers that need to be serviced

Each customer has a demand to be collected by the vehicle

Find a set of routes such that**:**

- Each route begins and ends at the depot
- Each customer is visited exactly once by exactly one route
- The total demand of the customers assigned to each route does not exceed Q
- The **total distance traveled by the vehicle is minimized**



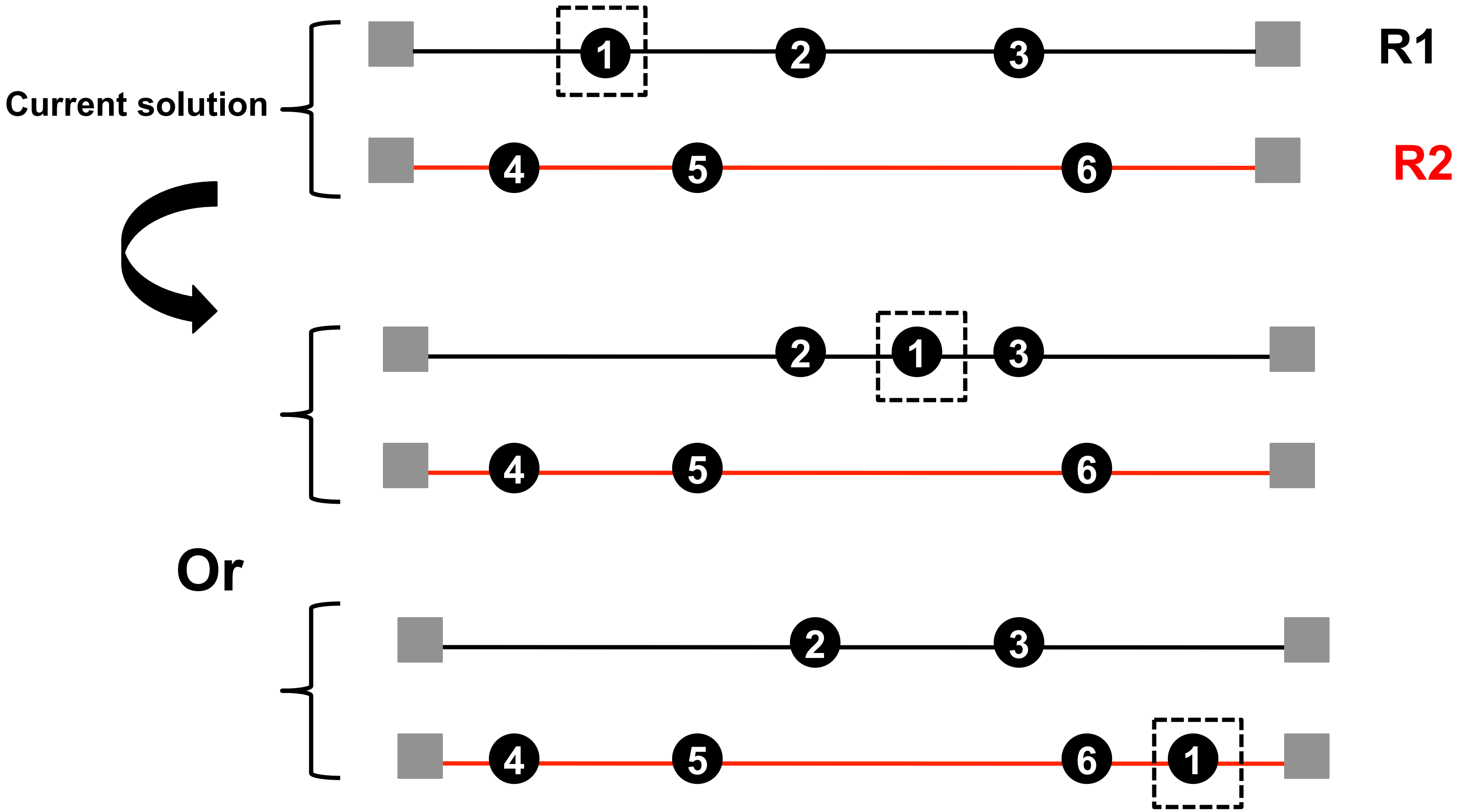
The **Stochastic** Vehicle Routing Problem

The **demand** of each customer is **uncertain**

Typically formulated as a **recourse model:**

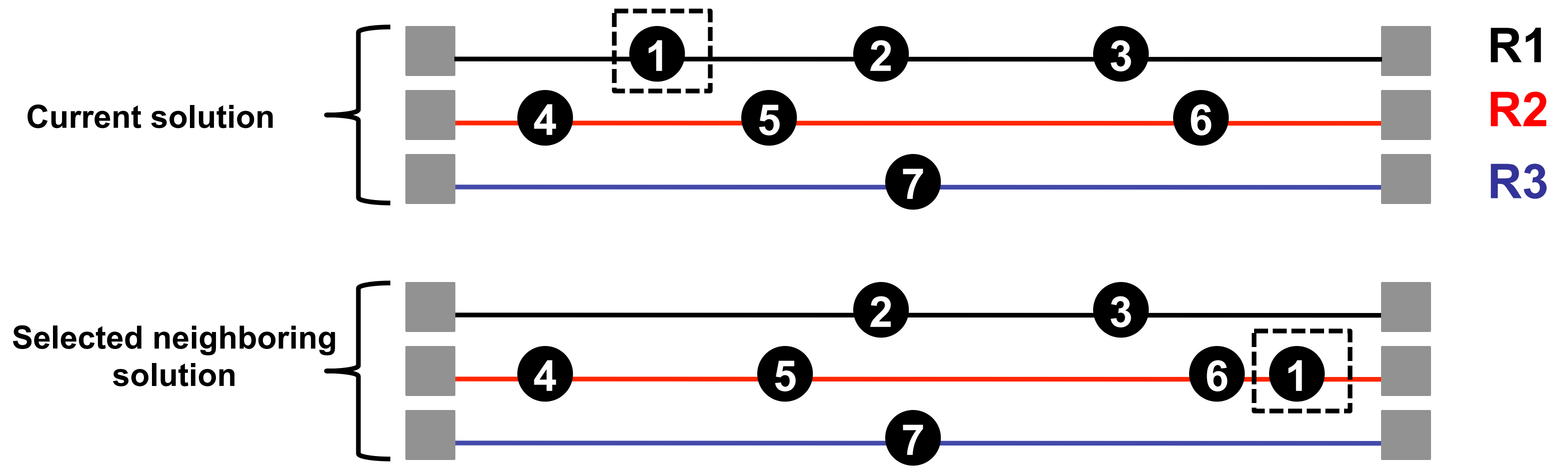
- Whenever the residual load of the vehicle reaches a specified threshold, the vehicle returns to the depot to unload and resumes collections at the next planned customer
- **Minimize the expected distance** traveled by the vehicle

A simple neighborhood



Provided that route R2 has sufficient residual capacity

Tabu list: some possibilities

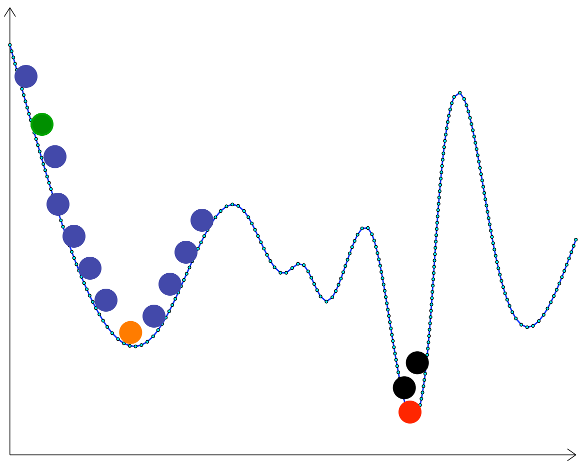


• Moving 1 back from R2 to R1 is Tabu Include (1, R2, R1) in the Tabu list	• Will not constrain the search much Cycling may occur if 1-->R3 then 1-->R1
• Moving 1 back to R1 is Tabu (without consideration for its current route) Include (1,R1) in the Tabu list	• Stronger May prohibit a solution with a value better than that of the best-known solution Aspiration criterion will allow the move
• Moving 1 to any route is Tabu Include (1) in the Tabu list	• Even stronger May lead to an overall stagnation of the searching process

Basic Tabu search is good but…

- "Too local": tends to spend most of its time in a restricted portion of the search space

- Although **good solutions** may be obtained, one may fail to explore the most interesting parts of the search space



- **Diversification**: a mechanism to force the search into previously unexplored parts of the search space

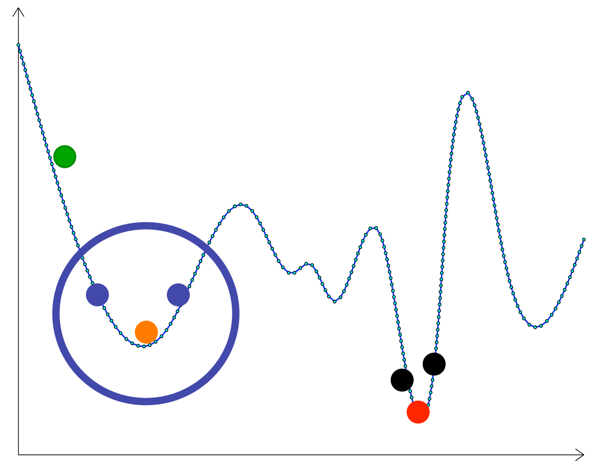
- Usually based on some form of **long-term memory** such as frequency memory

- Two major techniques
  - Continuous diversification
  - Restart diversification

Continuous diversification

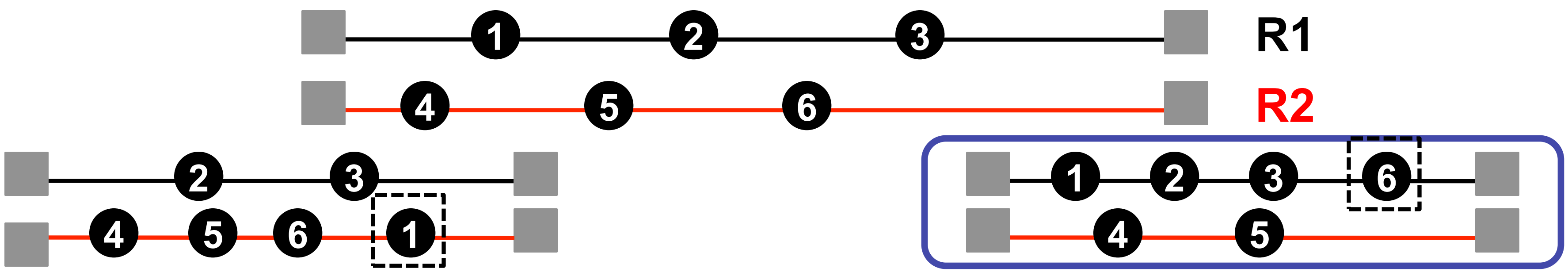
- Integrate diversification considerations into the searching process

- Bias the evaluation of possible moves by adding to the objective a small term related to the component frequencies (minimization problem)



Modified move value = Move value + diversification parameter * frequency measure

- A possible frequency measure in the SVRP application would be the number of times the customer involved in the move has been moved from its current route



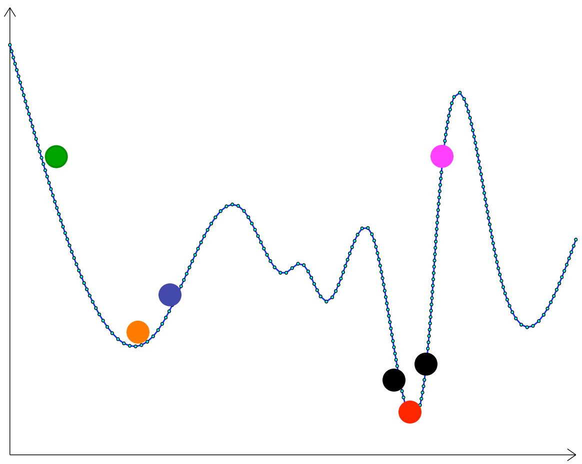
f(x1) = 25 and freq (1,R1) = 2

f(x2) = 26 and freq (6,R2) = 0

Diversification parameter = 0.75

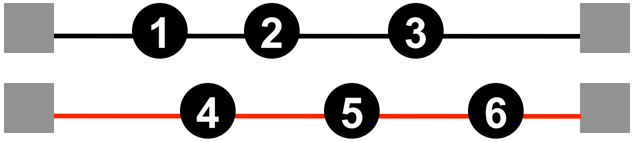
Restart diversification

- Force a few rarely used components in the **current solution** or the **best-known solution**
- Restart the search from **this point**

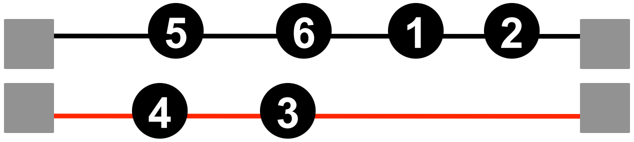


- A possible strategy in the SVRP application would be to force customers that have not yet been moved frequently into new routes

Current solution



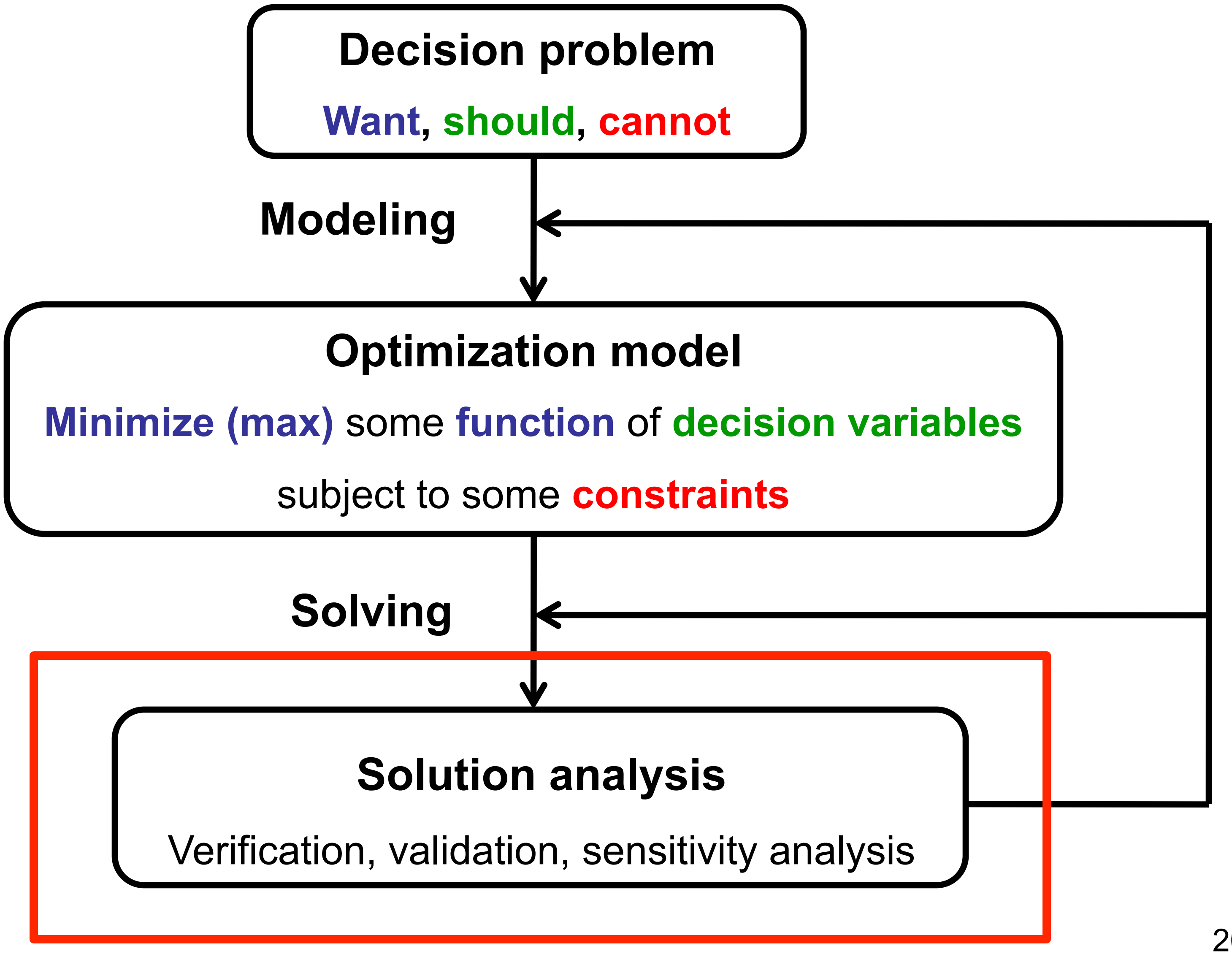
New initial solution



freq (1,R1) = 3, freq (1,R2) =2
freq (2,R1) = 3, freq (2,R2) =2
freq (3,R1) = 5, freq (3,R2) =0
freq (4,R1) = 2, freq (4,R2) =3
freq (5,R1) = 0, freq (5,R2) =5
freq (6,R1) = 1, freq (6,R2) =4

Current solution or best-known solution?
To what extent?
Feasibility?
Random, greedy?

The big picture



Evaluating metaheuristics

- **Quantitative criteria**
  - **Efficiency**: the method should provide **optimal or near-optimal** solutions to **realistic problem instances** (comparison with optimal solutions if known, with Lower/Upper bounds if known, or with other metaheuristics)
  - **Effectiveness**: **good solutions** should be achieved **within a reasonable amount of time**
  - **Robustness**: **efficiency and effectiveness** should prevail **for a variety of problem instances** (not just fine-tuned to some training set and less good elsewhere)
- **Qualitative criteria**
  - **Simplicity:** **simple** and **clear** principles should govern the method
  - **Flexibility:** the method should be **easily adapted** to deal with new problem variants
  - **User-friendliness**: the method should be easy to understand and **most important easy to use**. This implies it should have as few parameters as possible

Basic questions to ask when implementing Tabu Search

Tabu search
Initial solution
Neighborhood structure
Tabu (attributes, tenure, classification)
Aspiration criteria
Diversification strategy?
Stopping criteria

And then test and check ...

- Efficient?
- Effective?
- Robust?

A variant of the Stochastic OPMPS

- Determine the extraction order of blocks from a mineral deposit over a given time horizon

- Constraints

  Reserve constraints

  Slope constraints

  Mining constraints (lower and upper limits)

  Processing constraints (lower and upper limits)

  Metal production constraints (lower and upper limits)

- Objective function

  Maximize the expected NPV

  Minimize deviations from production targets

Basic questions to ask when implementing Tabu Search

Tabu search
Initial solution
Neighborhood structure
Tabu (attributes, tenure, classification)
Aspiration criteria
Diversification strategy?
Stopping criteria

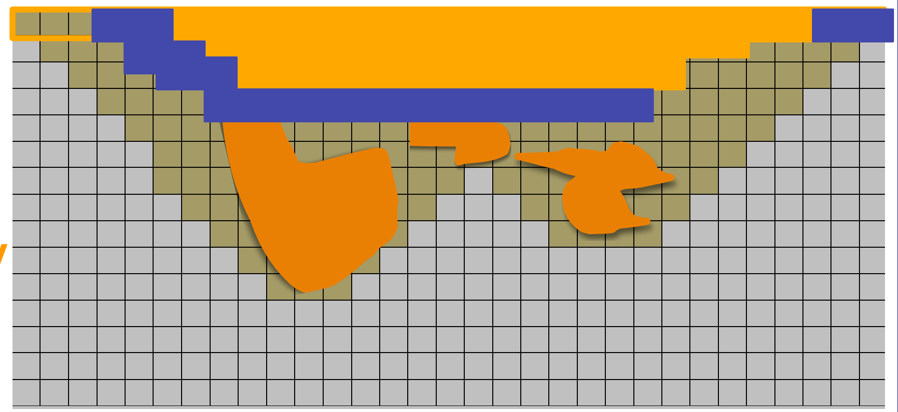
And then test and check ...

- Efficient?
- Effective?
- Robust?

Initial solution

Candidate blocks

Select a block randomly



A new period (t+1) is initiated whenever the total weight of the blocks extracted in t is greater or equal than 1/2(maxToExtract +minToExtract)

1	2	.....	i	...	j	....	k		N
2	2		1		1		T+1		T+1

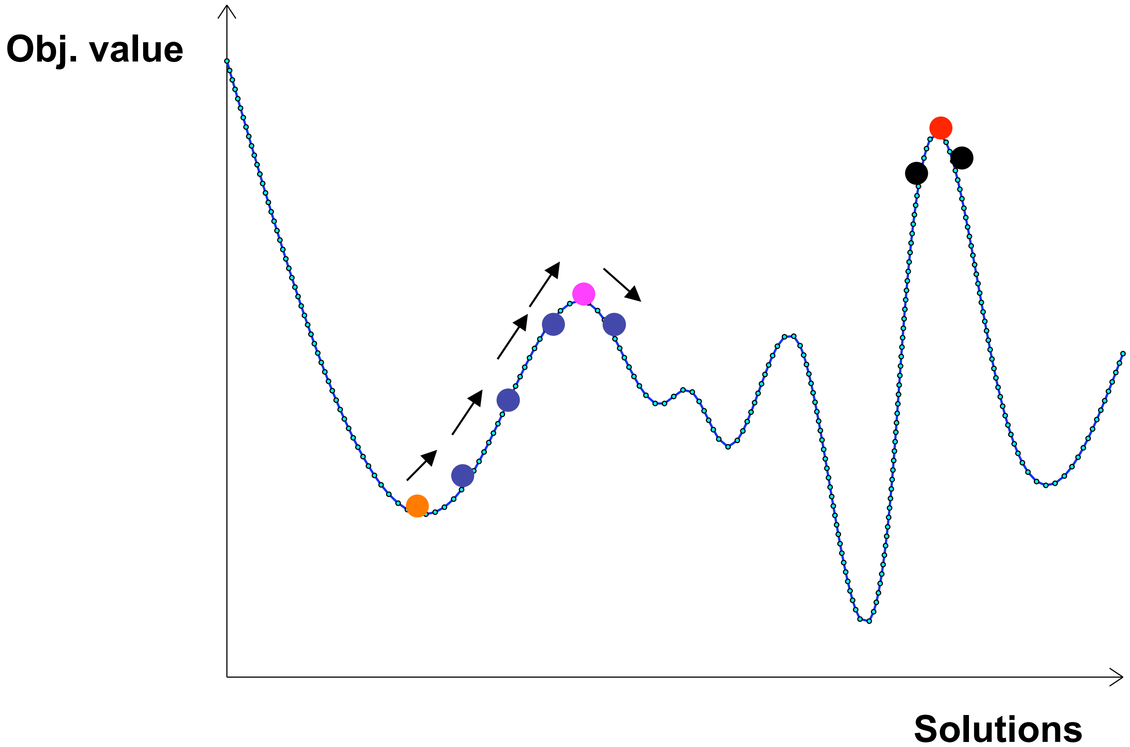
Basic questions to ask when implementing Tabu Search

Tabu search
Initial solution
Neighborhood structure
Tabu (attributes, tenure, classification)
Aspiration criteria
Diversification strategy?
Stopping criteria

And then test and check ...

- Efficient?
- Effective?
- Robust?

Neighborhood Structure



Neighborhood

Shift block *i* from period t_1 to another period t_2

The solution generated belongs to the neighborhood if it is feasible

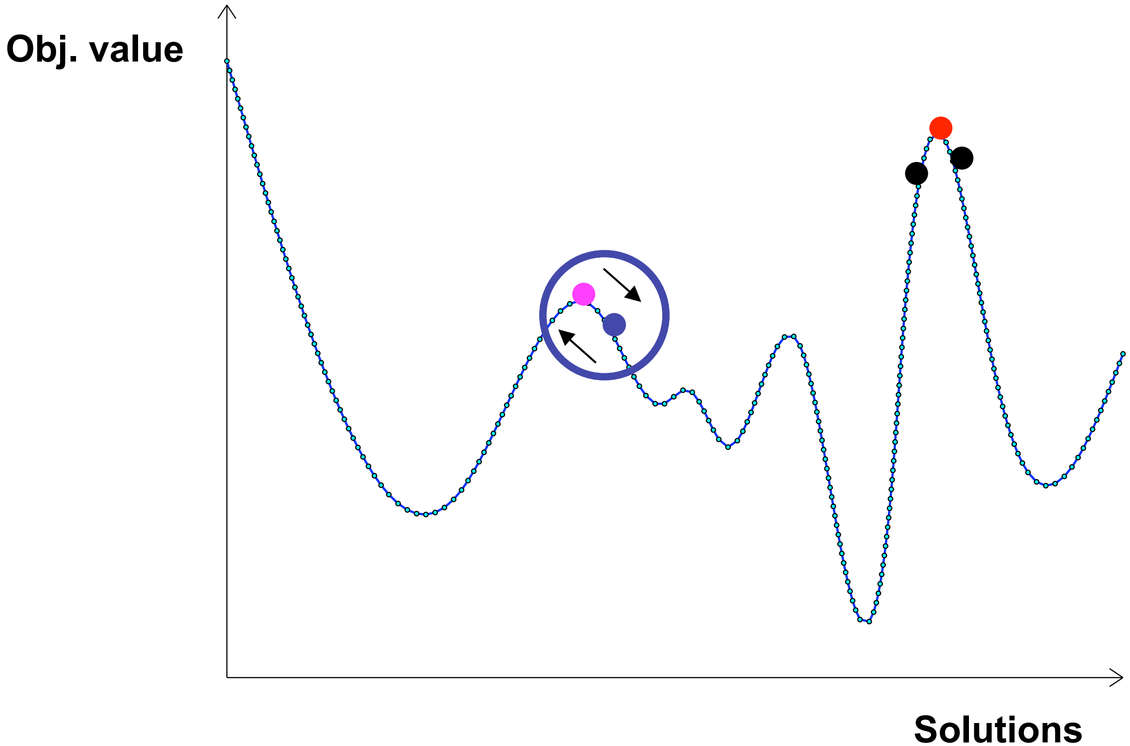
Basic questions to ask when implementing Tabu Search

Tabu search
Initial solution
Neighborhood structure
Tabu (attributes, tenure, classification)
Aspiration criteria
Diversification strategy?
Stopping criteria

And then test and check ...

- Efficient?
- Effective?
- Robust?

Tabu and aspiration criteria



Do not allow reversing **recent** shifts (are declared Tabu for a certain number of iterations)

Except if they lead to a solution better than the best solution found so far (aspiration)

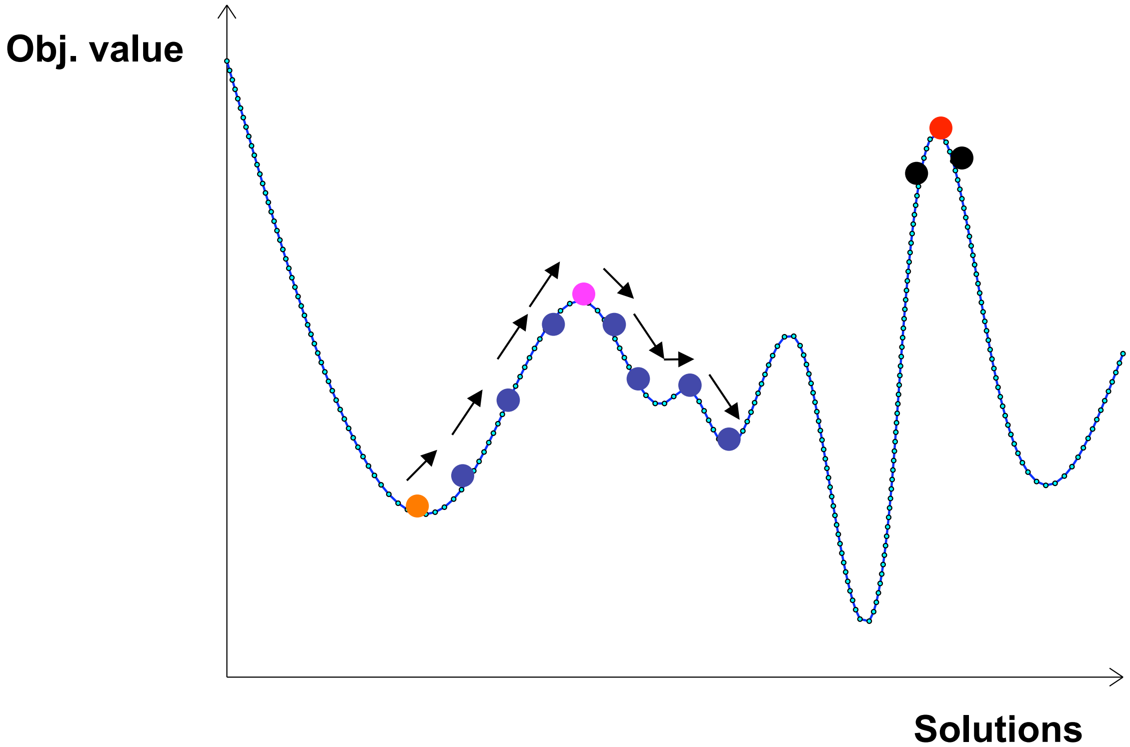
Basic questions to ask when implementing Tabu Search

Tabu search
Initial solution
Neighborhood structure
Tabu (attributes, tenure, classification)
Aspiration criteria
Diversification strategy?
Stopping criteria

And then test and check ...

- Efficient?
- Effective?
- Robust?

Stopping criterion



nitermax successive iterations where the objective does not improve

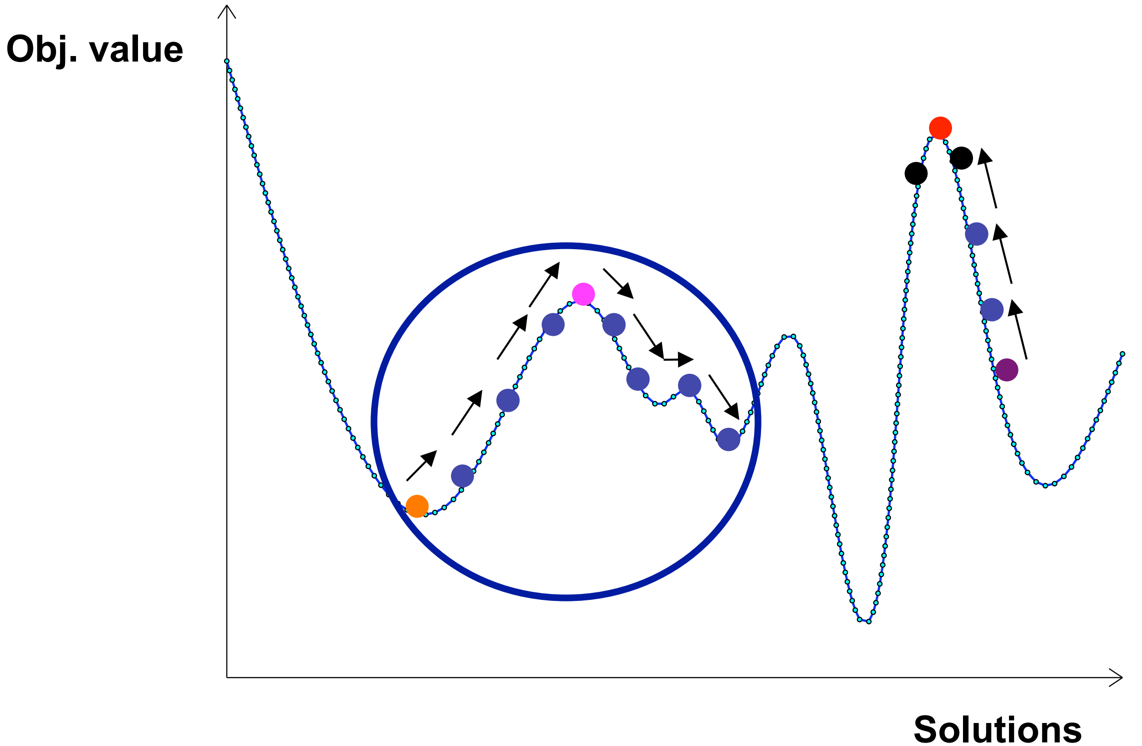
Basic questions to ask when implementing Tabu Search

Tabu search
Initial solution
Neighborhood structure
Tabu (attributes, tenure, classification)
Aspiration criteria
Diversification strategy?
Stopping criteria

And then test and check ...

- Efficient?
- Effective?
- Robust?

Diversification strategy



Restart the search from a **new point** **(more extensive search)**

Apply successively a sequence of shifts in order to generate **the new initial solution**

Diversification strategy

- Apply successively a sequence of shifts in order to generate **the new initial solution**
- Applied to

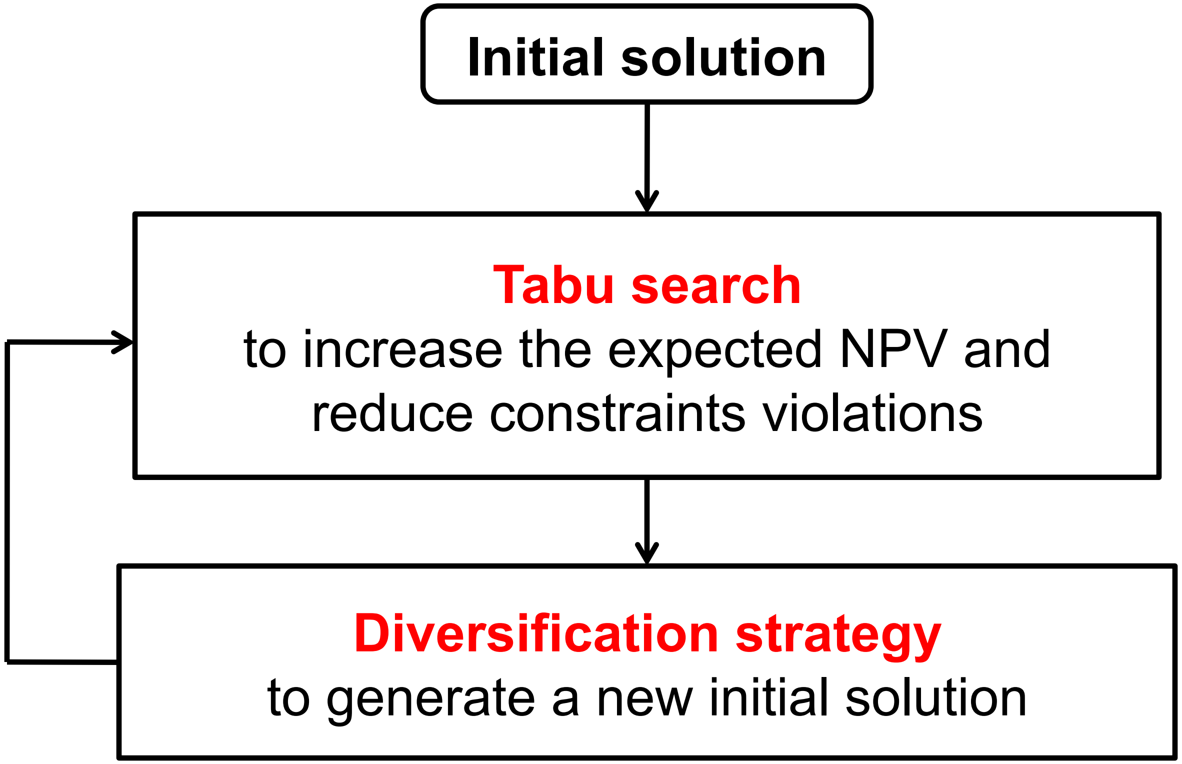
TS	VNS
Current best solution (generated during the last search)	**Best-known solution (found so far)**

- Choose the shifts

  Trying to move the search towards unexplored or less explored areas of the solution space

  Trying to obtain a good quality solution

Solution procedure



Repeat as long as the elapsed time is less than a specified maximum time

Basic questions to ask when implementing Tabu Search

Tabu search
Initial solution
Neighborhood structure
Tabu (attributes, tenure, classification)
Aspiration criteria
Diversification strategy?
Stopping criteria

And then test and check ...

- Efficient?
- Effective?
- Robust?

Numerical results

- 2 sets of problems $\mathbf{P_1}$, $\mathbf{P_2}$. In each set, 5 problems having different sizes

  $\mathbf{P_1}$: from a copper deposit

  $\mathbf{P_2}$: from a gold deposit

- Each problem is solved 10 times using different initial solutions

Set	Problem	Number of blocks (N)	Number of periods (T)
P_1 Metal type: copper Block size: $20 \times 20 \times 10\,m$ Block weight: $w_i = 10,800$ tons	C1	4,273	3
	C2	7,141	4
	C3	12,627	7
	C4	20,626	10
	C5	26,021	13
P_2 Metal type: gold Block size: $15 \times 15 \times 10\,m$ Block weight: $w_i = 5,625$ tons	G1	18,821	5
	G2	23,901	7
	G3	30,013	8
	G4	34,981	9
	G5	40,762	11

Basic questions to ask when implementing Tabu Search

Tabu search
Initial solution
Neighborhood structure
Tabu (attributes, tenure, classification)
Aspiration criteria
Diversification strategy?
Stopping criteria

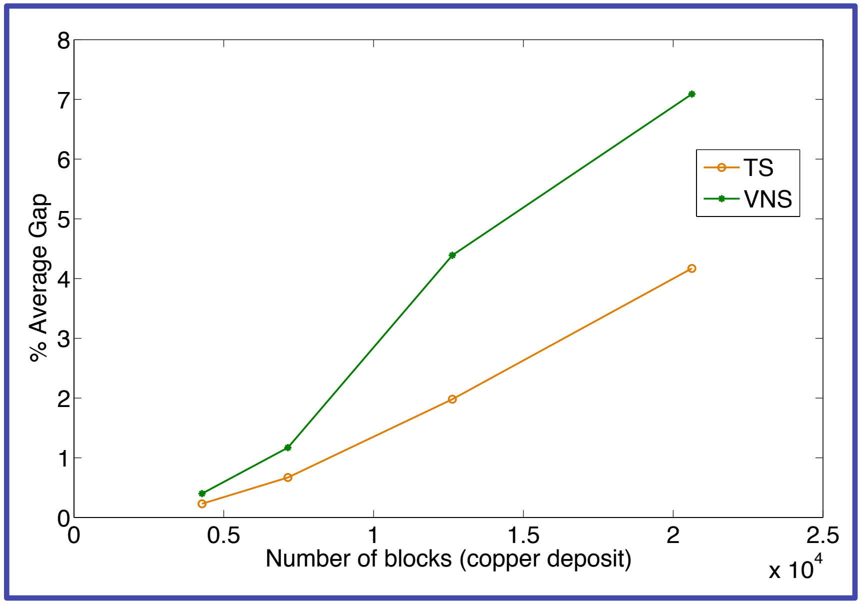
And then test and check ...

- Efficient?
- Effective?
- Robust?

- **Efficiency: the method should provide optimal or near-optimal solutions to realistic problem instances (comparison with optimal solutions if known, with Lower/Upper bounds if known, or with other metaheuristics)**
- **Effectiveness: good solutions should be achieved within a reasonable amount of time**
- **Robustness: efficiency and effectiveness should prevail for a variety of problem instances (not just fine-tuned to some training set and less good elsewhere)**

Numerical results: Copper deposit

$$\%Average\,Gap = \frac{Z_{LR} - Z_{average}}{Z_{LR}} \times 100$$



Worth using metaheuristics

CPLEX: much more CPU

C5 (N=26,021, T=13): fails to solve the LR within 4 weeks VS 2 hours
C4 (N=20,626, T=10): 9 days VS 1 hour

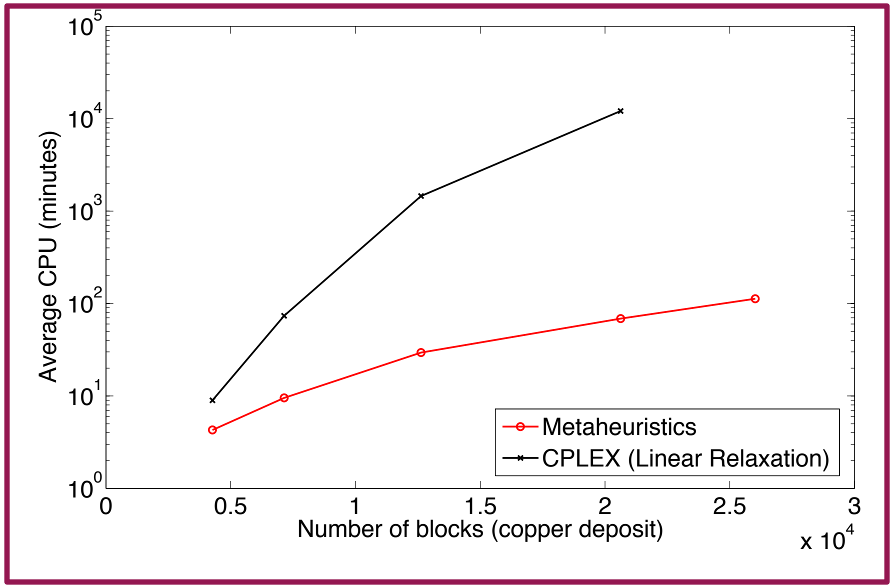
Metaheuristics: small gap

In average: < 4% (1.76% and 3.26%)

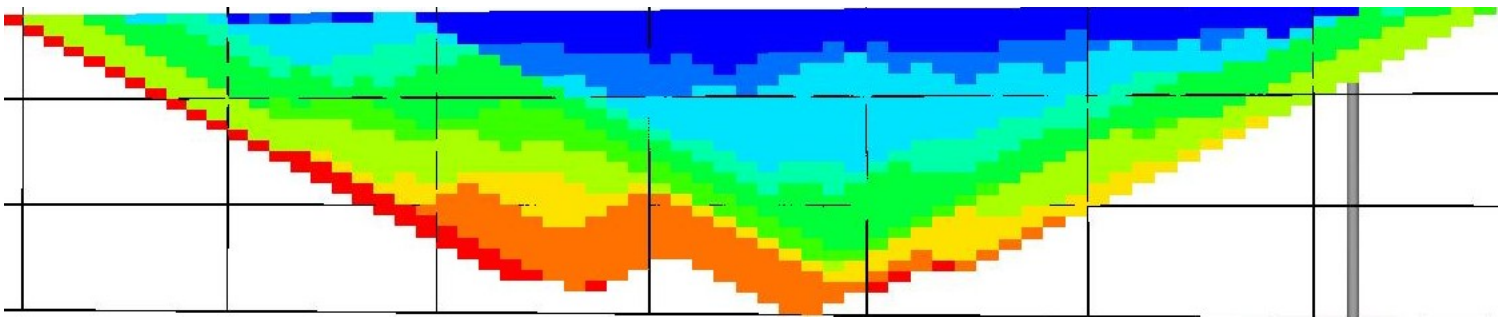
TS outperforms VNS

Improves %Gap over VNS by 46%

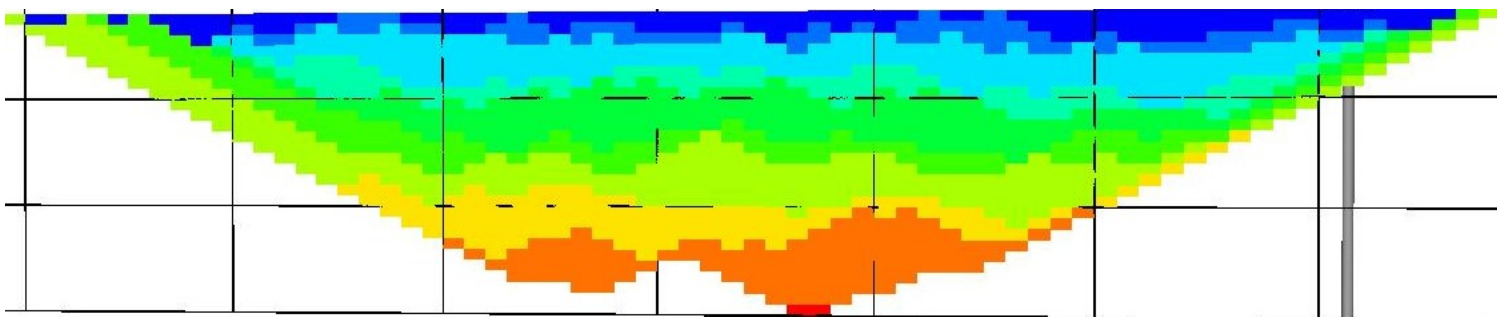
Performance differences increase with the problem size



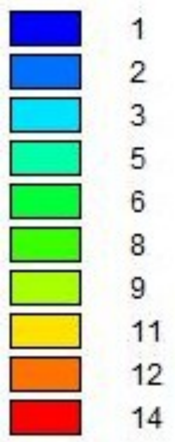
Numerical results: Problem C5



Cross-sectional view of the best schedule generated by TS



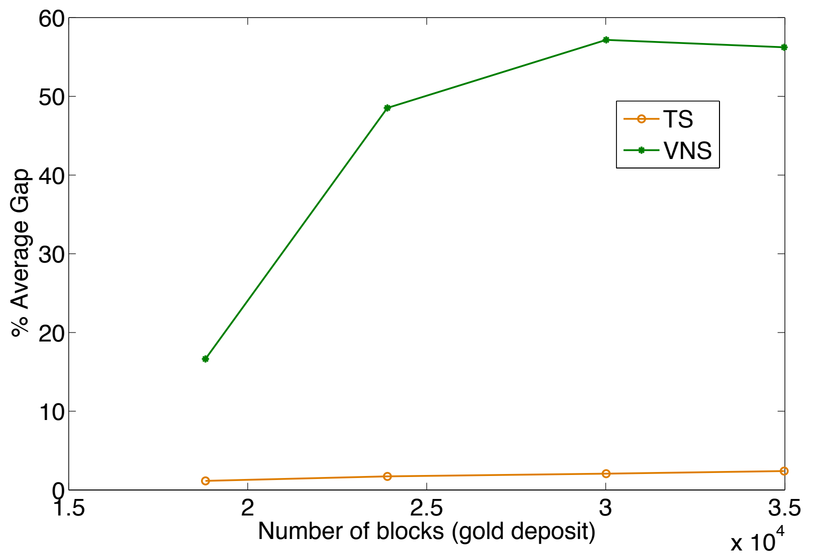
Cross-sectional view of the best schedule generated by VNS



TS improves the value of objective function over VNS by 20%

Numerical results: Gold deposit

$$\% Average\, Gap = \frac{Z_{LR} - Z_{average}}{Z_{LR}} \times 100$$



Worth using metaheuristics

CPLEX: much more CPU

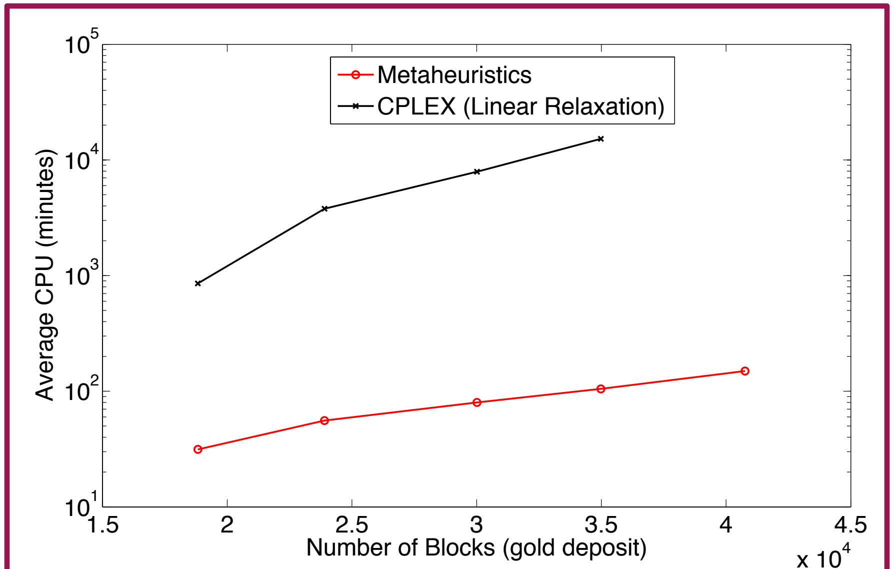
G5 (N=40,762, T=11): fails to solve the LR within 4 weeks VS 2.5 hours
G4 (N=34,981, T=9): 11 days VS 2 hours

TS: very small gap
In average: < 2% (1.83%)

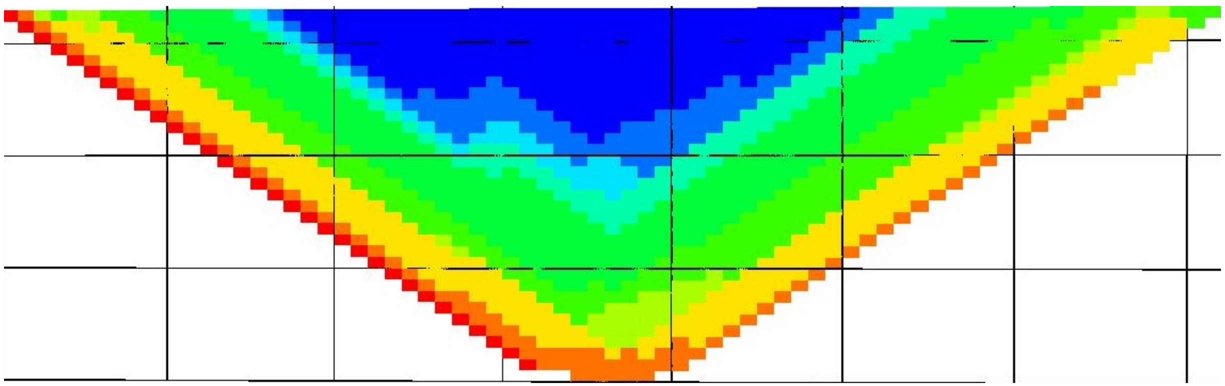
TS outperforms VNS

Improves %Gap over VNS by 96%

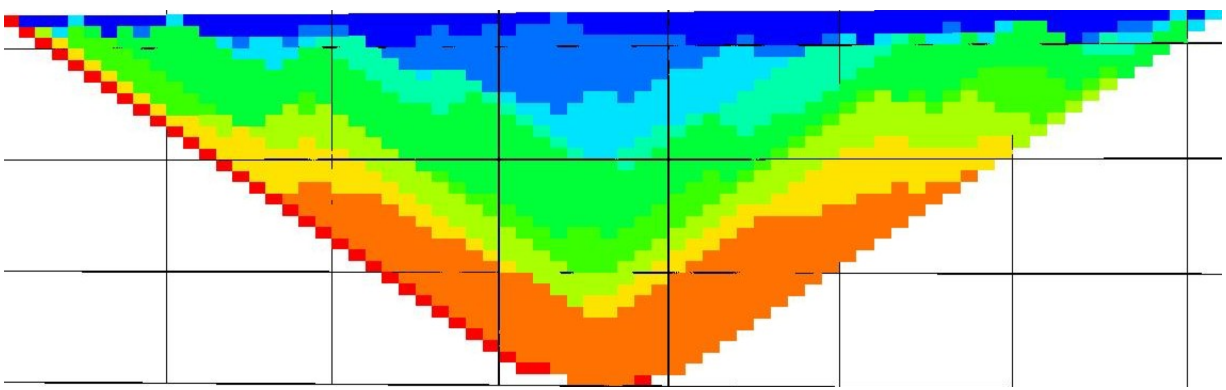
Differences more pronounced
%Gap in [1.15, 2.40] VS [16.64, 57.17]



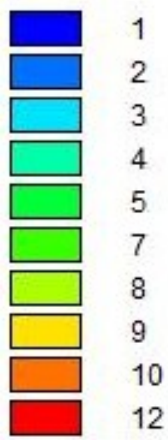
Numerical results: Problem G5



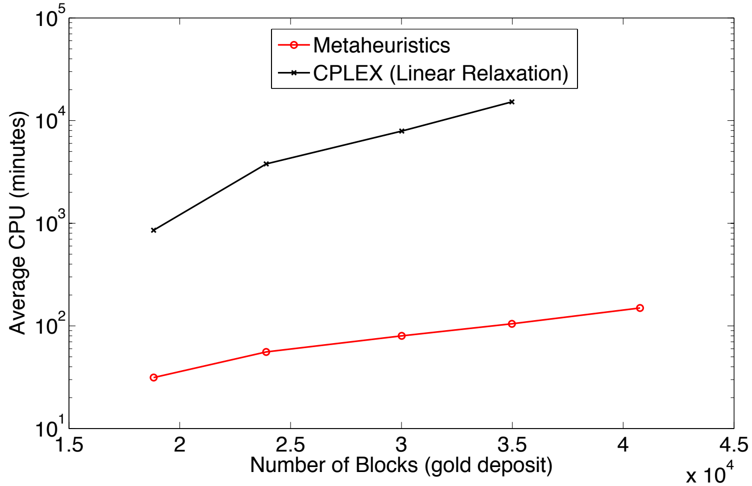
Cross-sectional view of the best schedule generated by TS

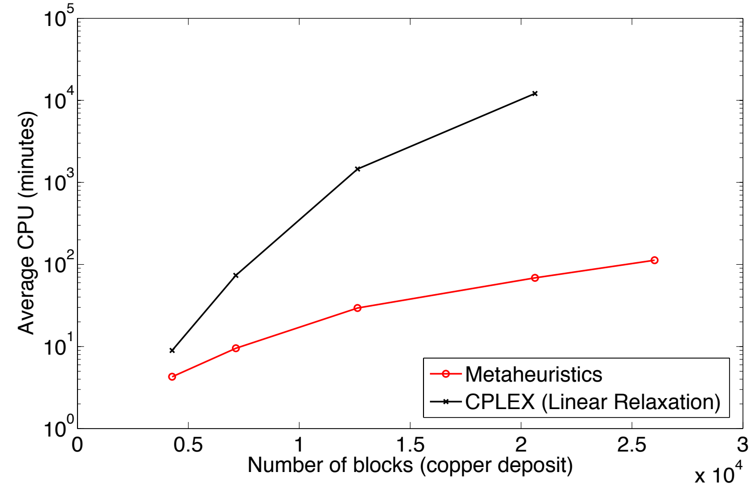


Cross-sectional view of the best schedule generated by VNS

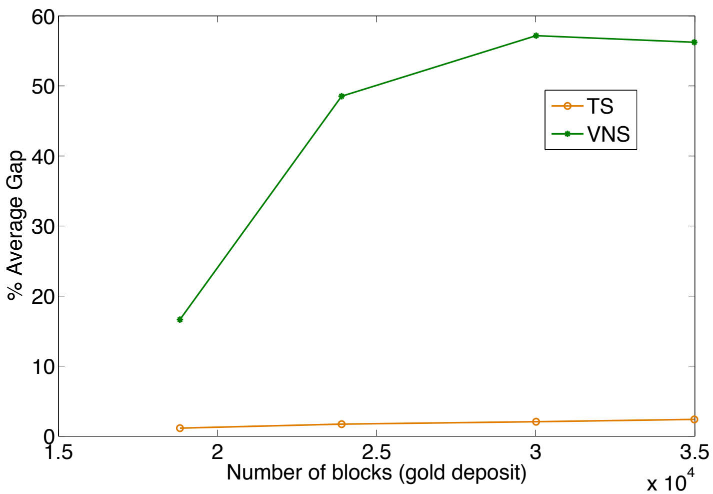


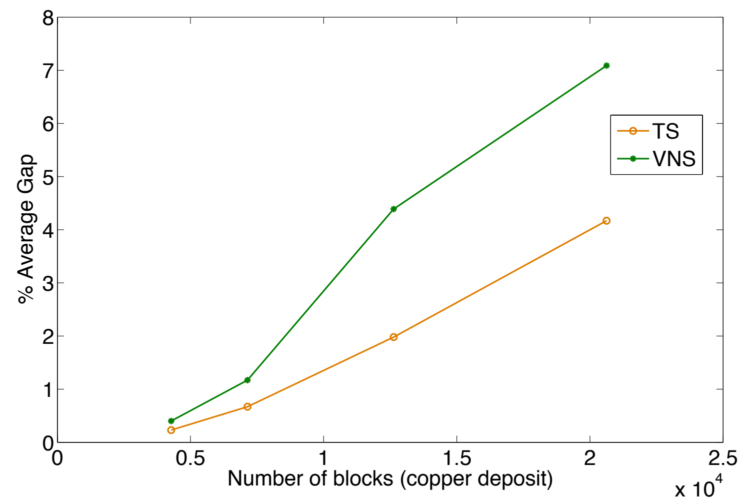
TS improves the value of objective function over VNS by 126%





TS and VNS require significantly shorter CPU than CPLEX (LR)





TS more effective and more robust than VNS

Conclusions

- Metaheuristics are powerful algorithmic approaches which have been applied with great success to may difficult optimization problems including Stochastic Combinatorial optimization problems
- They allow dealing with large size problems having high degree of complexity, and generate rapidly very good solutions
- Optimality is not guaranteed in general and few convergence results are known for special cases

Conclusions

Tabu search
Initial solution
Neighborhood structure
Tabu (attributes, tenure, classification)
Aspiration criteria
Diversification strategy?
Stopping criteria

Further readings:

- Glover F. and Laguna M., “Tabu Search”, Kluwer Academic Publishers, Boston, 1997
- Glover F. and Kochenberg G.A., “Handbook of metaheuristics”, Kluwer Academic Publishers, Boston, 2003

Conclusions

- Metaheuristics cannot be applied blindly to any problem. Generally, **ad hoc adaptations** are used to deal with specific applications

  Significant knowledge and understanding of the problem at hand is absolutely required to develop a successful metaheuristic implementation