Are you a Hybrid? Yes, of course, everyone is a Hybrid nowadays!

Christian Blum

Artificial Intelligence Research Institute (iiia)
Spanish National Research Council (csic)

IIIA-CSIC

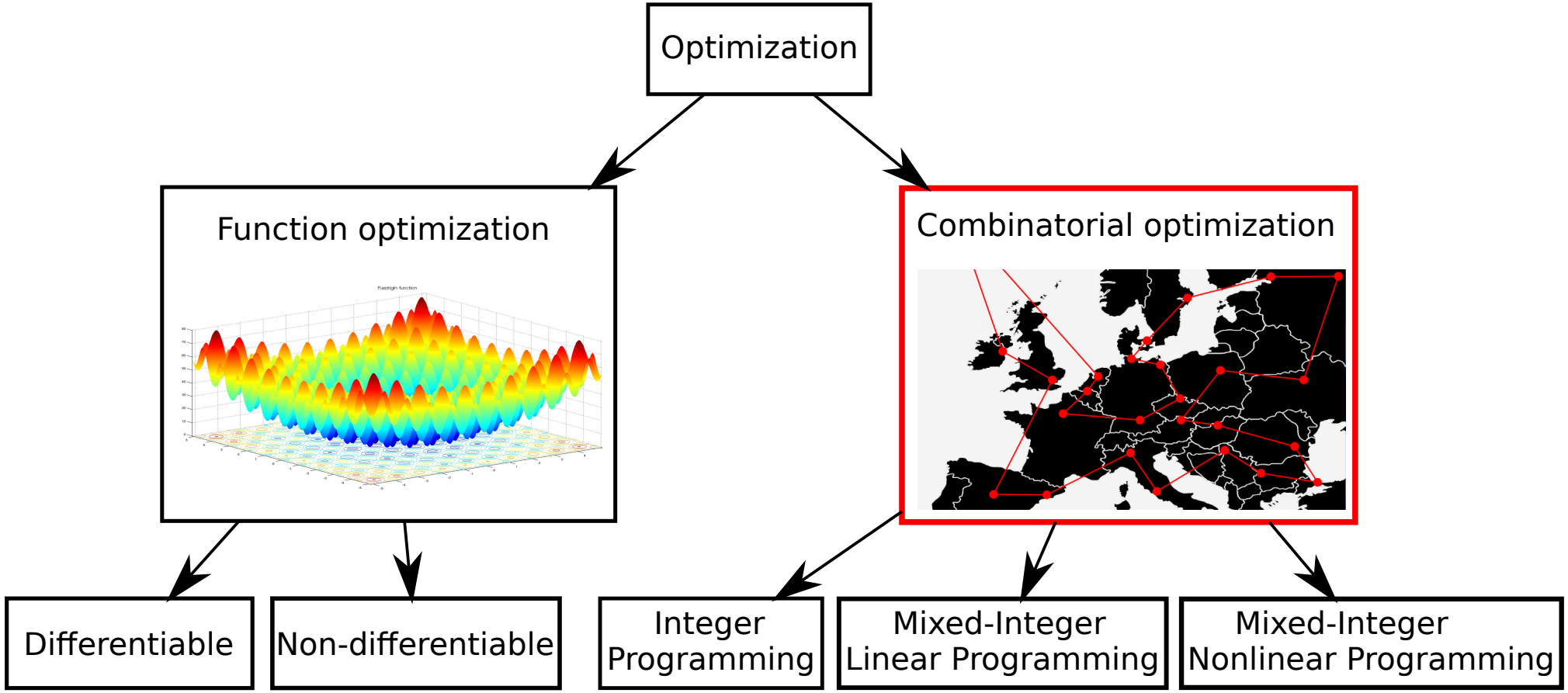
CSIC: Spanish National Research Council

- Largest public institution dedicated to research in Spain (created in 1939)
- Third-largest in Europe
- 6% of all research staff in Spain work for the CSIC
- 20% of the scientific production in Spain is from the CSIC

IIIA: Artificial Intelligence Research Institute

- 20 tenured scientists (of three different ranks)
- Around 35 additional staff member (post-docs, PhD students, technicians, administration)
- Three research lines (machine learning, logic and constraint programming, multi-agent systems)

Topic of today: preparing the grounds

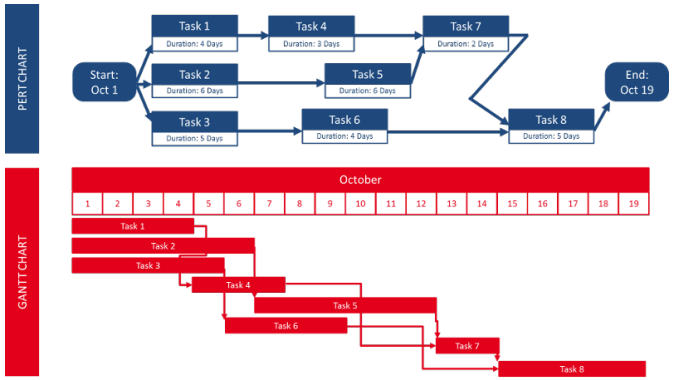


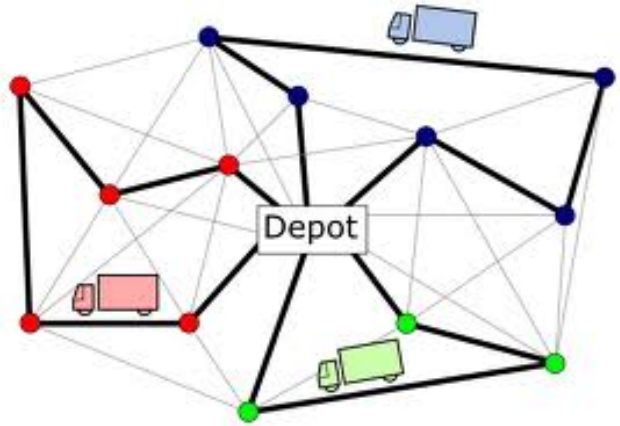
Importance of combinatorial optimization problems

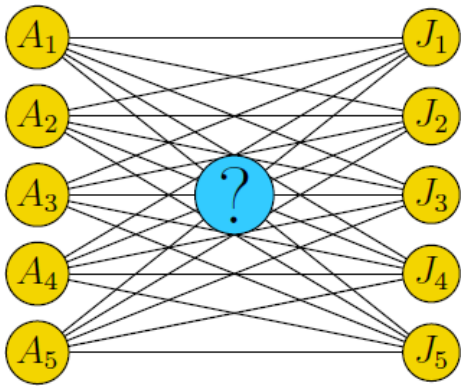
Note: Combinatorial optimiztion problems arise in numerous industrial settings

Examples

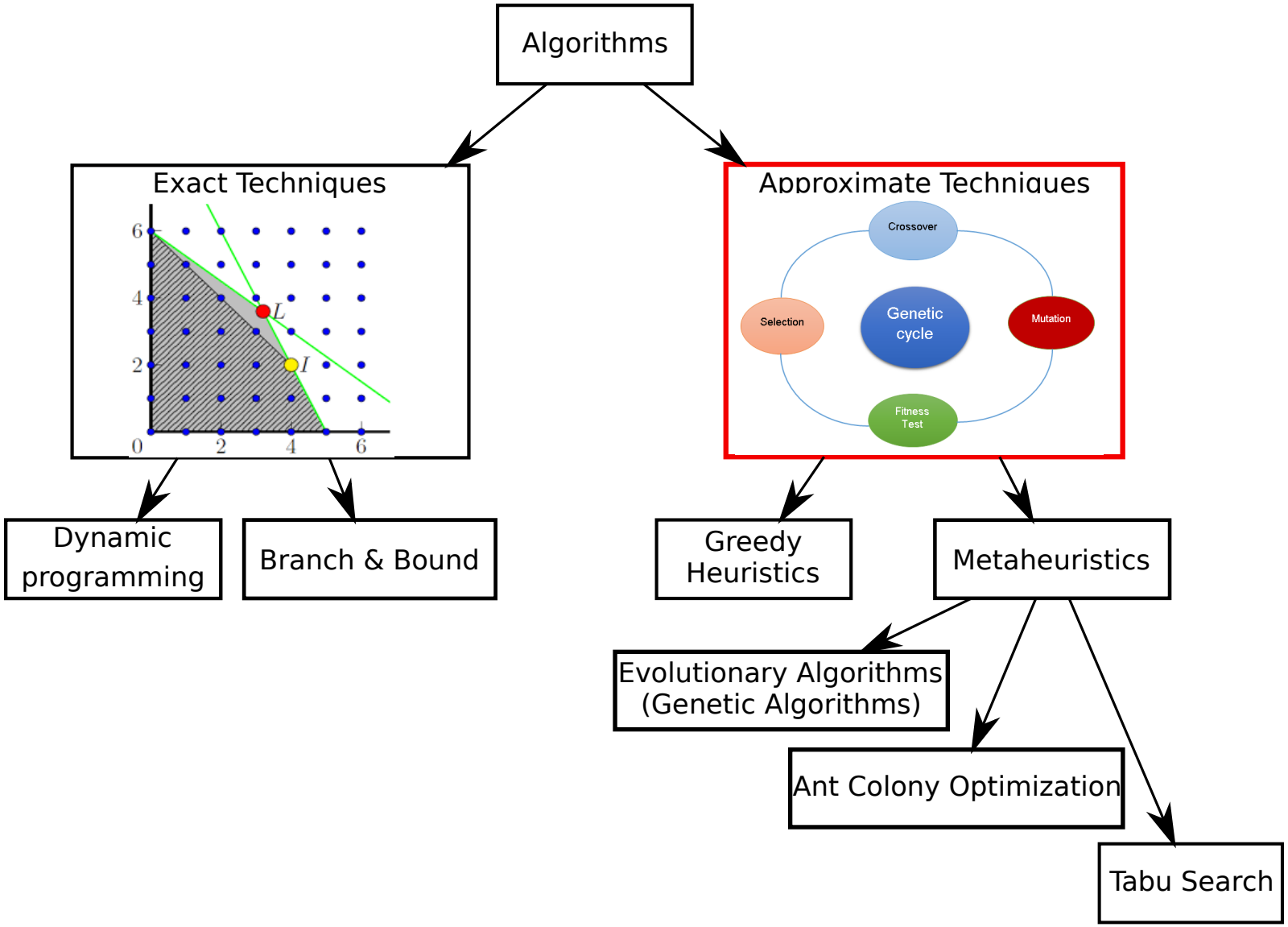
- Routing problems
- Assignment problems
- Scheduling problems



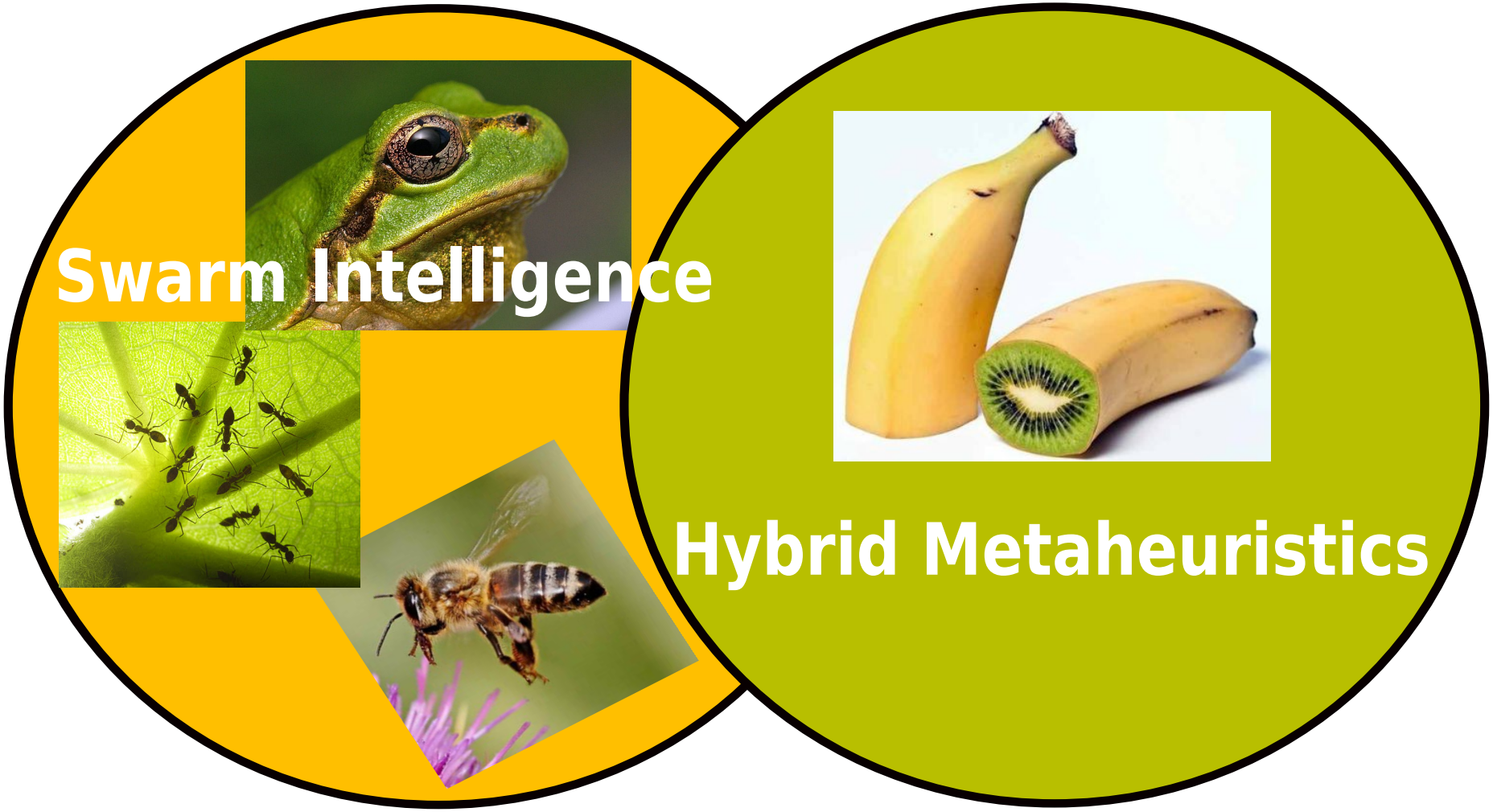




Algorithms for combinatorial optimization

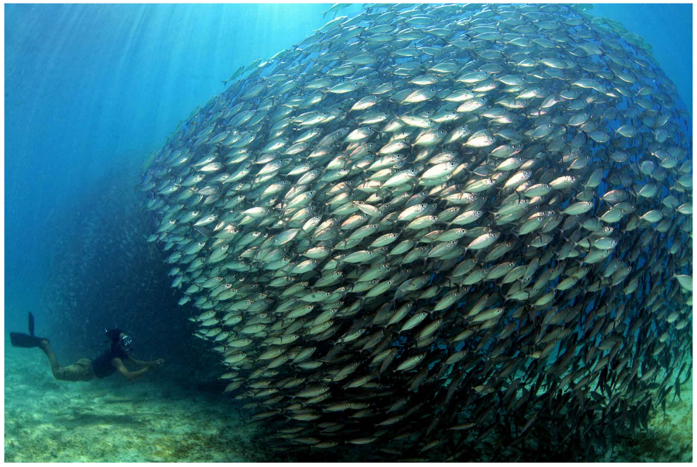


My research topics: algorithm-oriented (basic research)



What is swarm intelligence?

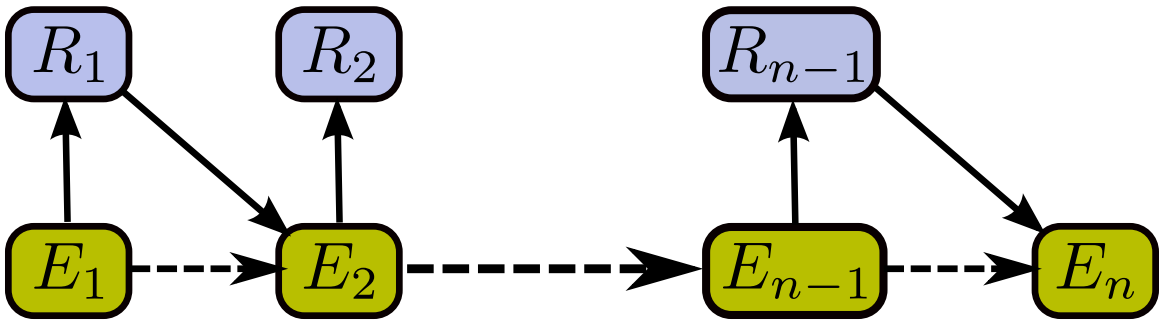
In a nutshell: AI discipline whose goal is designing intelligent multi-agent systems by taking inspiration from the collective behaviour of animal societies such as ant colonies, flocks of birds, or fish schools



Swarm intelligence

Properties:

- Consist of a set of simple entities
- Distributedness: No global control
- Self-organization by:
  - ⋆ **Direct communication:** for example, by visual or chemical contact
  - ⋆ **Indirect communication:** Stigmergy (Grassé, 1959)



Result: Complex tasks/behaviors can be accomplished/exhibited in cooperation

Swarm Intelligence topics from last years

- Combinatorial optimization: adding negative learning to ant colony optimization
  Inspiration: foraging behaviour of ant colonies

- Distributed optimization: graph coloring, independent set finding
  Inspiration: self-desynchronization in Japanese tree frogs

- Distributed problem solving: duty-cycling in sensor networks
  Inspiration: work-synchronization in ant colonies

More info: On my website

`https://www.iiia.csic.es/~christian.blum/`

Hybrid metaheuristics: definition

Definition: What is a hybrid metaheuristic?

- Problem: a precise definition is not possible/desirable

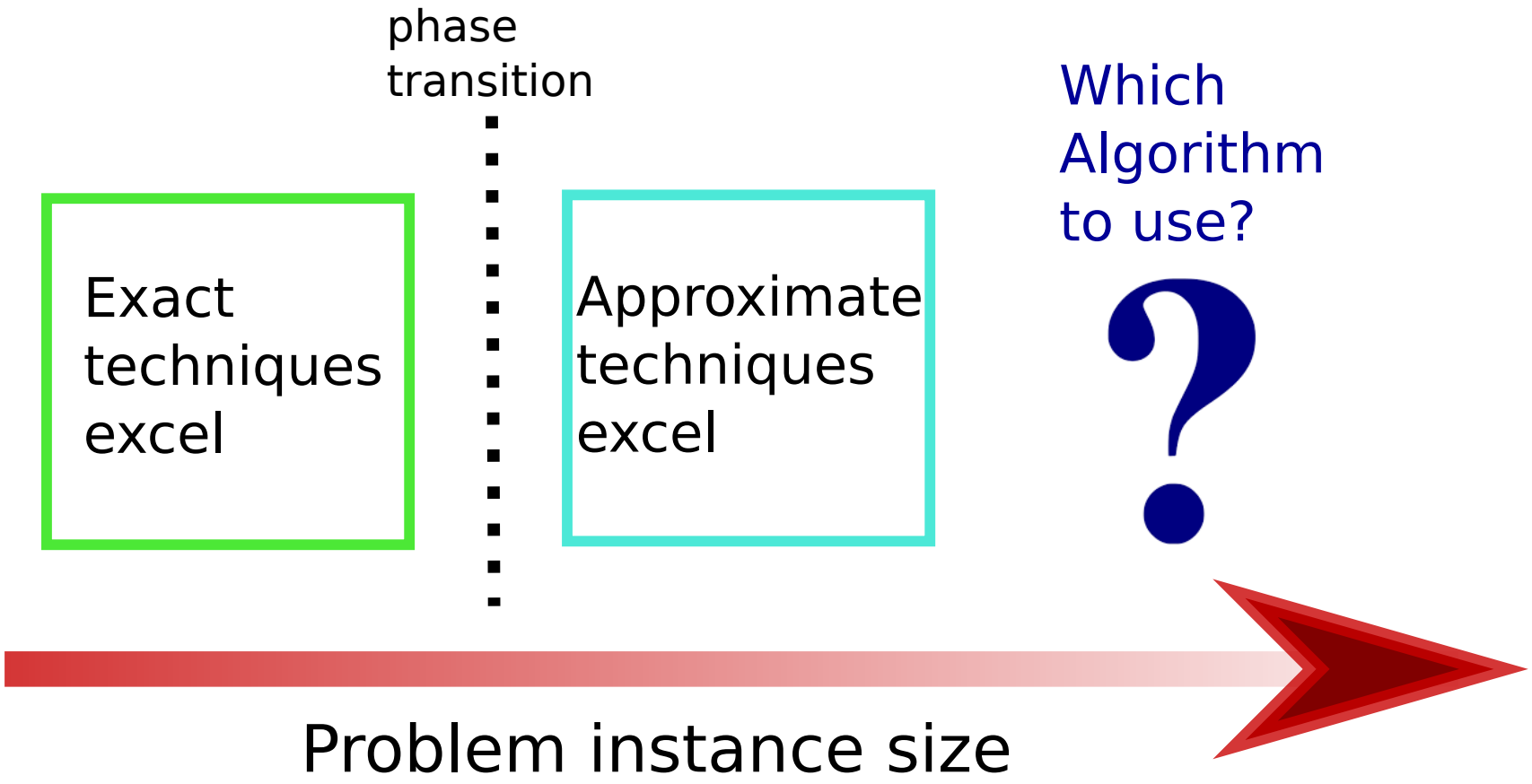
Possible characterization:

A technique that results from the combination of a metaheuristic with other techniques for optimization

What is meant by: other techniques for optimization?

- Metaheuristics
- Branch & bound
- Dynamic programming
- Integer Linear Programming (ILP) techniques

Hybrid metaheuristics: reason of being



Note: Hybrid algorithms that exploit the synergies between exact and approximate algorithms often excel in the context of large-scale problem instances

Hybrid metaheuristics: history

History:

- For a long time the different communities co-existed quite isolated
- Hybrid approaches were developed already early, but only sporadically
- Only since about 15 years the published body of research grows significantly:
  1. 1999: CP-AI-OR Conferences/Workshops
  2. 2004: Workshop series on Hybrid Metaheuristics
  3. 2006: Matheuristics Workshops

Consequence: The term hybrid metaheuristics identifies a seperate line of research

Example: decoder-based approaches (1)

General idea:

- Represent feasible solutions to the problem in an indirect way
- Design a decoder that translates such a solution into an actual solution

Advantages:

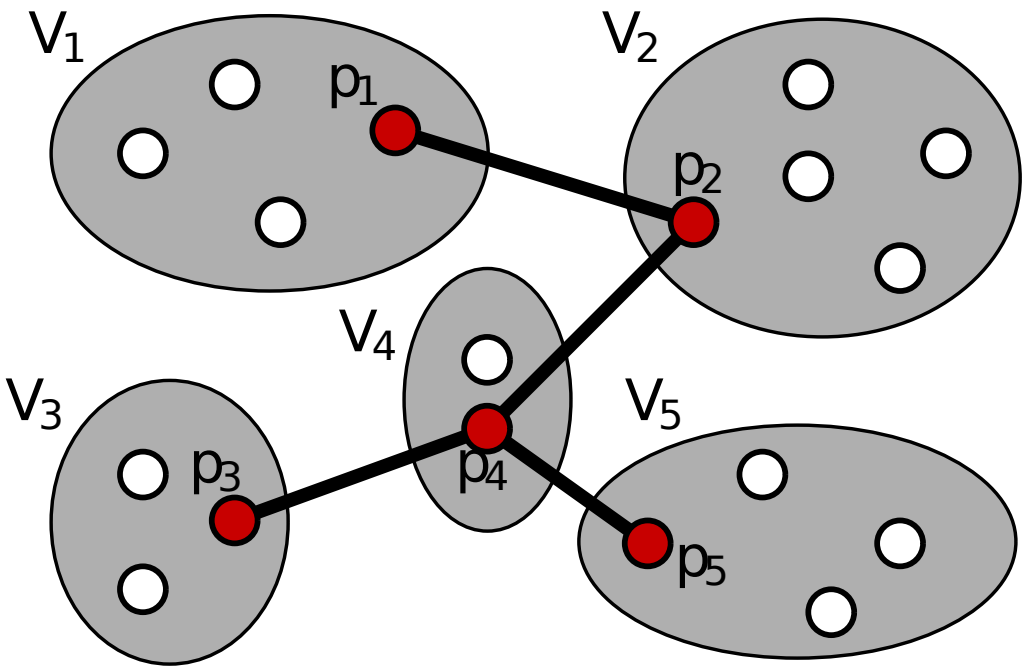
- This may transform a complex search space into one that is easier to handle
- Encapsulates problem complexity into the decoder

Note:

- Already very well known in the field of evolutionary algorithms
- Example: biased random key genetic algorithms (BRKGAs)

Example: decoder-based approaches (2)

Case study: Hu, B. et al. (2008). **Combining variable neighborhood search with integer linear programming for the generalized minimum spanning tree problem**. **Journal of Heuristics**, 14(5), 473-499.



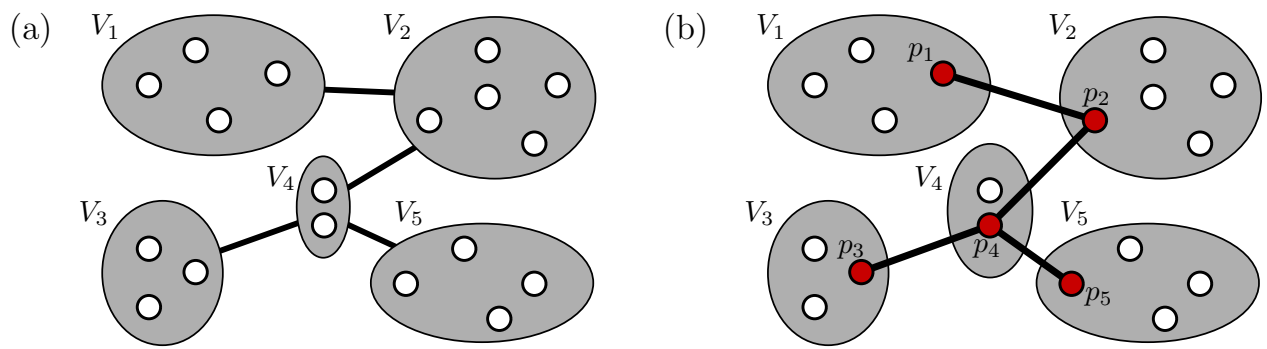
Example: decoder-based approaches (3)

Option 1:

- Define a feasible solution as a set of nodes which contains exactly one node from each cluster
- Obtain an actual solution by applying Prim's algorithm to a feasible solution

Option 2:

- Define a feasible solution as a spanning tree of the clusters
- Utilize a dynamic programming algorithm to know which node to choose from each cluster



Example: multi-level framework (1)

Springer Link

Published: October 2004

Multilevel Refinement for Combinatorial Optimisation Problems

Chris Walshaw

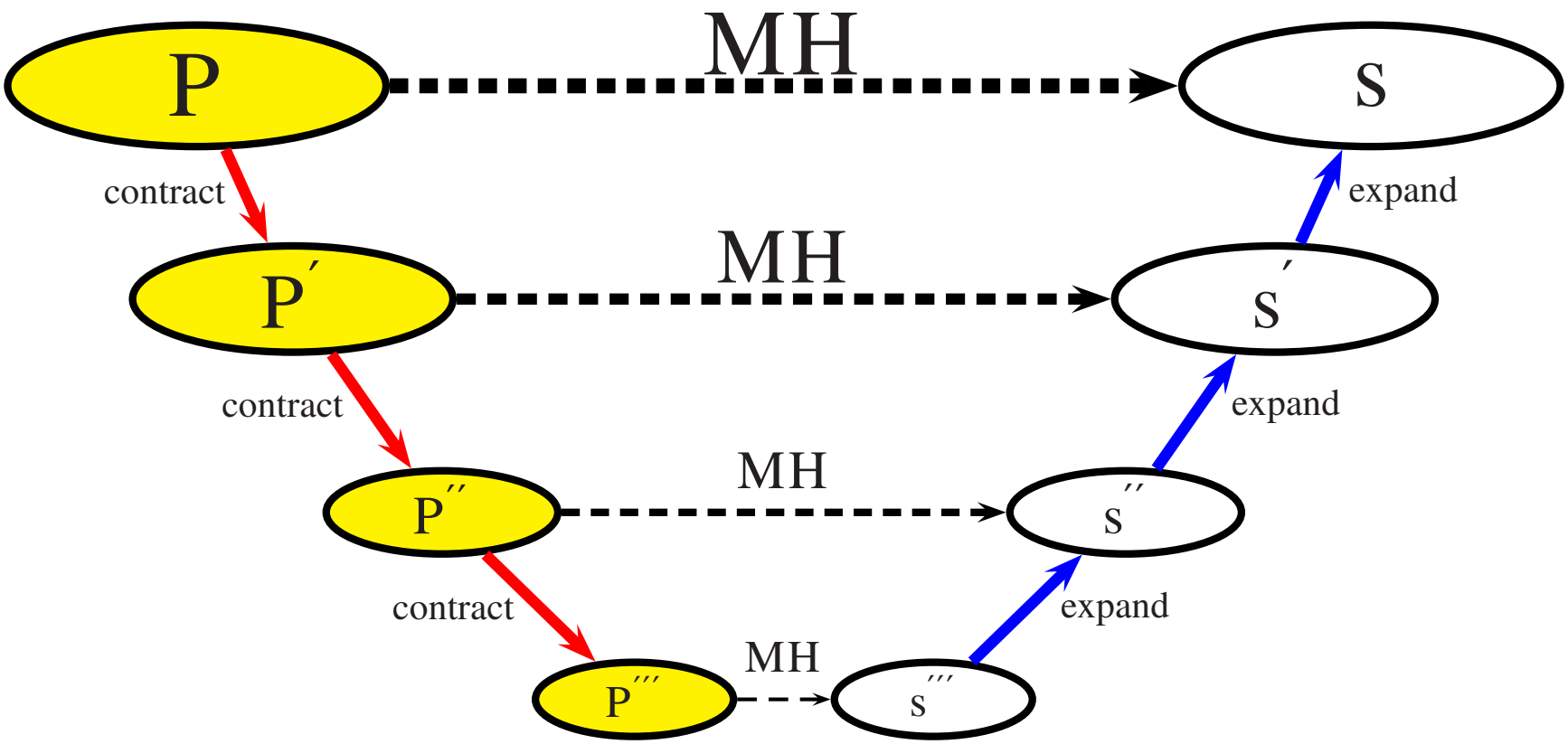
Annals of Operations Research **131**, 325–372(2004) | Cite this article

General idea:

- **First:** Iterative coarsening of the original problem instance
- **Then:** Find a solution to the coarsest level
- **Finally:** Iteratively refine this solution at each level

Example: multi-level framework (2)

The multi-level framework:



Example: multi-level framework (3)

ELSEVIER

Knowledge-Based Systems
Volume 195, 11 May 2020, 105678

A coarsening method for bipartite networks via weight-constrained label propagation ☆

Alan Valejo [a], Thiago Faleiros [b], Maria Cristina Ferreira de Oliveira [a], Alneu de Andrade Lopes [a]

SIAM Journal on Scientific Computing

< Previous Article Volume 41, Issue 4 Next Article >

Abstract | References | PDF | Cited By

SIAM J. Sci. Comput., 41(4), A2117–A2145. (29 pages)

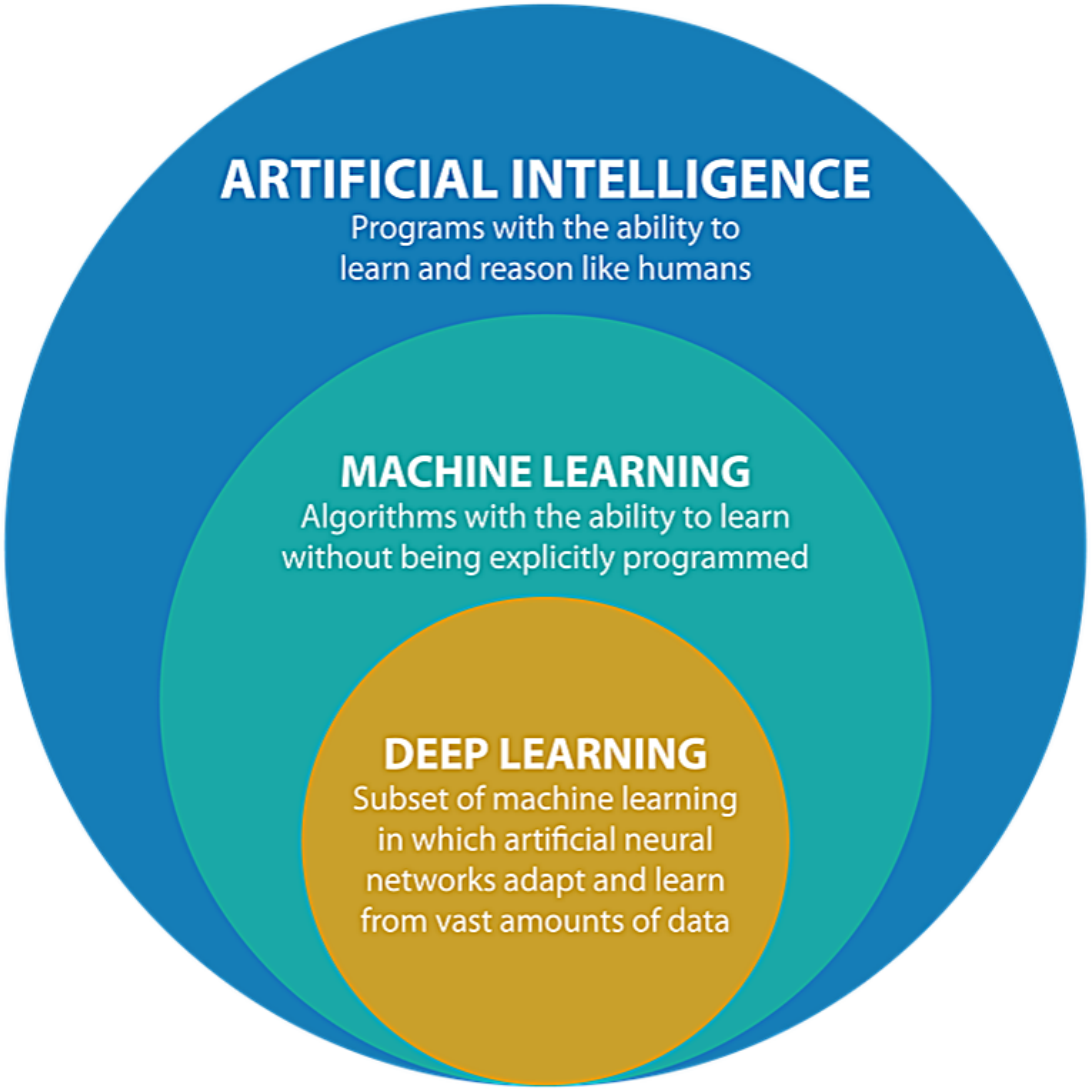
Methods and Algorithms for Scientific Computing

Multilevel Algorithms for Acyclic Partitioning of Directed Acyclic Graphs

Julien Herrmann, M. Yusuf Özkaya, Bora Uçar, Kamer Kaya, and Ümit V. Çatalyürek

Related Databases

Example: combinations with machine learning (1)



- Machine learning:
  - ⋆ Reinforcement learning
- Deep learning:
  - ⋆ Recurrent neural networks
  - ⋆ Convolutional neural networks

Example: combinations with machine learning (2)

ELSEVIER

European Journal of Operational Research

Available online 8 August 2020

In Press, Corrected Proof

Invited Review

Machine learning for combinatorial optimization: A methodological tour d'horizon

Yoshua Bengio [c, b], Andrea Lodi [a, b], Antoine Prouvost [a, b]

Note: machine learning may be used in (at least) two ways for improving algorithms for combinatorial optimization:

1. Generate a fast approximation for heavy computational tasks
2. Learning missing expert knowledge (for example a greedy function)

Example: combinations with machine learning (3)

Case study: learning greedy policy E. Khalil et al. (2017). **Learning combinatorial optimization algorithms over graphs**. In: Advances in Neural Information Processing Systems (**NIPS**) (pp. 6348-6358).

Used methodology:

- Deep learning architecture `structure2vec`.
- Training: type of Q-learning

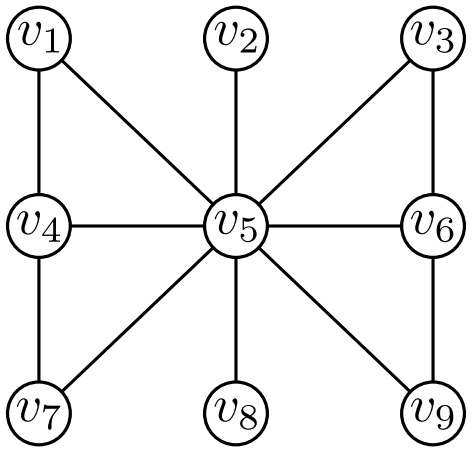
Results: The learned greedy policies ...

- outperform classical greedy algorithms for Minimum Vertex Cover, Maximum Cut and Traveling Salesman problem
- generalize to different problem instance sizes

Example: combinations with machine learning (4)

Note: Such a learned greedy function can be very interesting for problems for which no well-working greedy function is known.

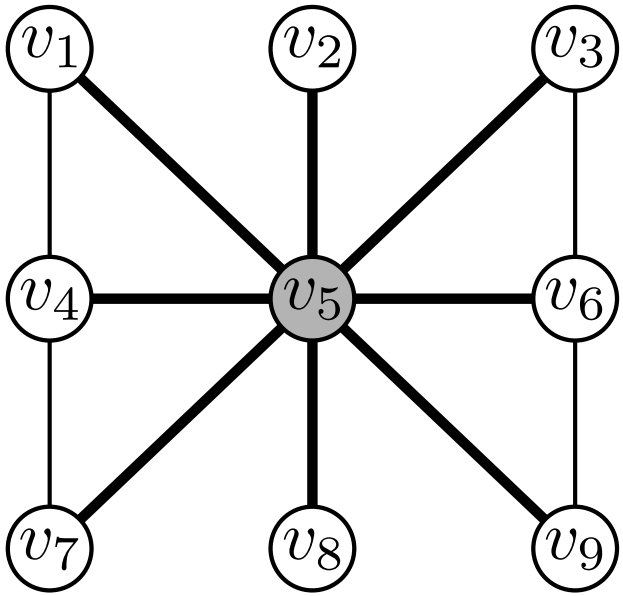
Example: Capacitated minimmum dominating set (CapMDS) problem



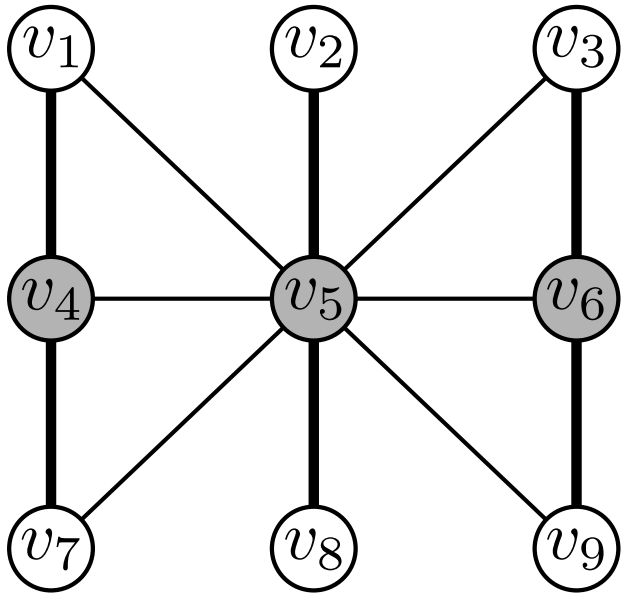
Classical MDS: Find the smallest subset of nodes such that each node of the graph (1) forms part of the subset or (2) is a neighbor of at least one node in the subset.

Example: combinations with machine learning (5)

CapMDS additional constraint: Each node can only cover a limited number of neighbors



Optimal MDS solution



Optimal CapMDS solution

(when max. 2 neighbors can be covered)

More Detailed Example

Algorithms Based on Problem Instance Reduction

Main idea: apply an exact technique to sub-instances

Subset selection problems

Note: For simplicity, algorithms in this part are explained in the context of subset selection problems.

Keep in mind: These algorithms can be applied to any combinatorial problem.

Definition: **Subset Selection Problem** (C, F, f)

1. C is a finite set of n items.
2. $F : 2^C \mapsto \{\text{TRUE}, \text{FALSE}\}$ indicates for each subset $S \subseteq C$ if it is feasible solution. Let $X \subseteq 2^C$ be the set of all feasible solutions.
3. $f : X \mapsto \mathbb{R}$ is an objective function that assigns a value to each feasible solution.

Note: Many well-known combinatorial optimization problems can be expressed as subset selection problems (Example: TSP).

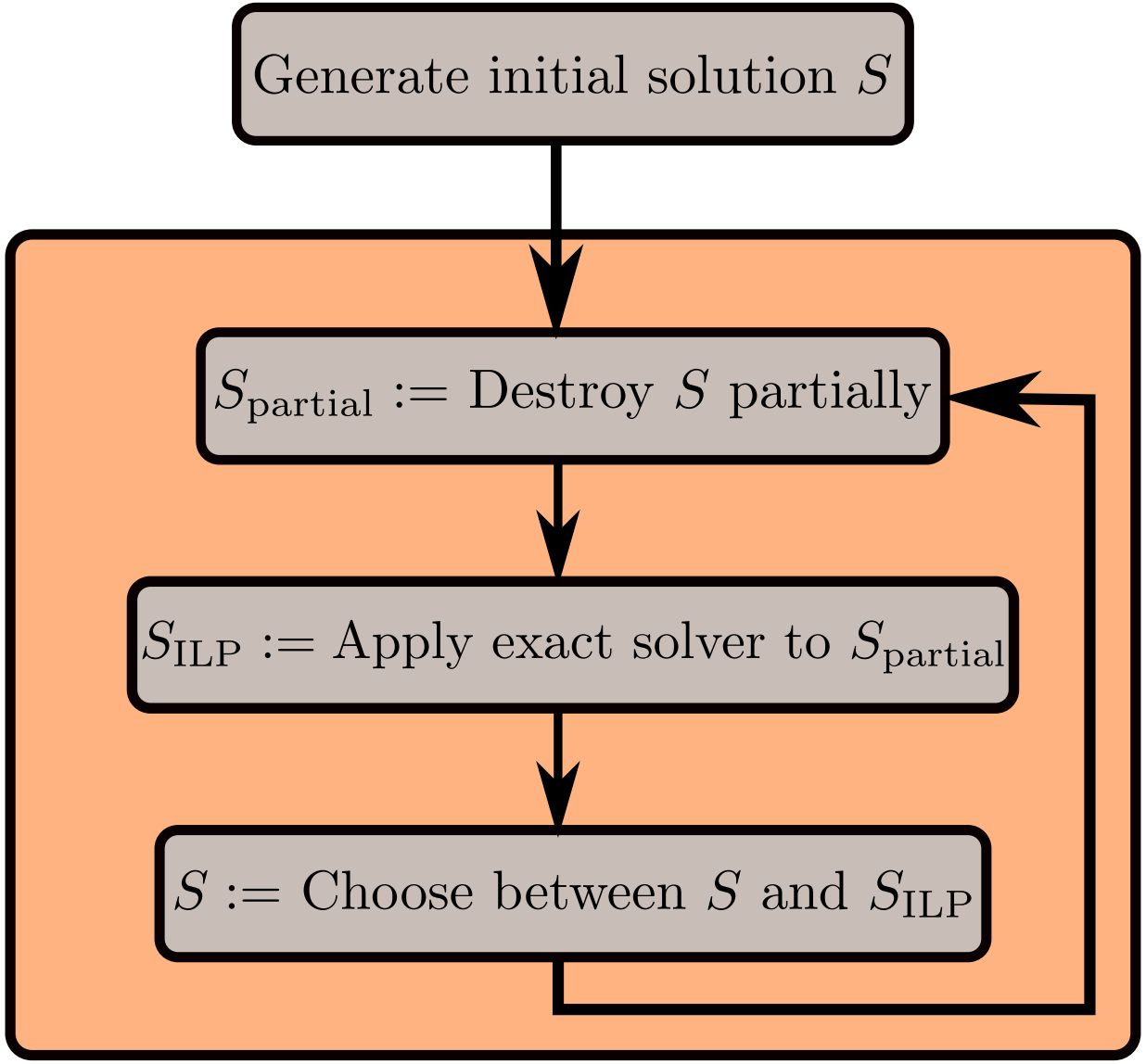
Standard: Large Neighborhood Search (LNS)

- Small neighborhoods:
  1. **Advantage:** It is fast to find an improving neighbor (if any)
  2. **Disadvantage:** The average quality of the local minima is low
- Large neighborhoods:
  1. **Advantage:** The average quality of the local minima is high
  2. **Disadvantage:** Finding an improving neighbor might itself be NP-hard due to the size of the neigbhorhood

Ways of examining large neighborhoods:

- Heuristically
- Exact techniques: for example an ILP solver

Destruction-based large neighborhood search



Alternative: Construct, Merge, Solve & Adapt (CMSA)

ELSEVIER

Computers & Operations Research
Volume 68, April 2016, Pages 75-88

Construct, Merge, Solve & Adapt A new general algorithm for combinatorial optimization

Christian Blum [a, b], Pedro Pinacho [a, c], Manuel López-Ibáñez [d], José A. Lozano [a]

ELSEVIER

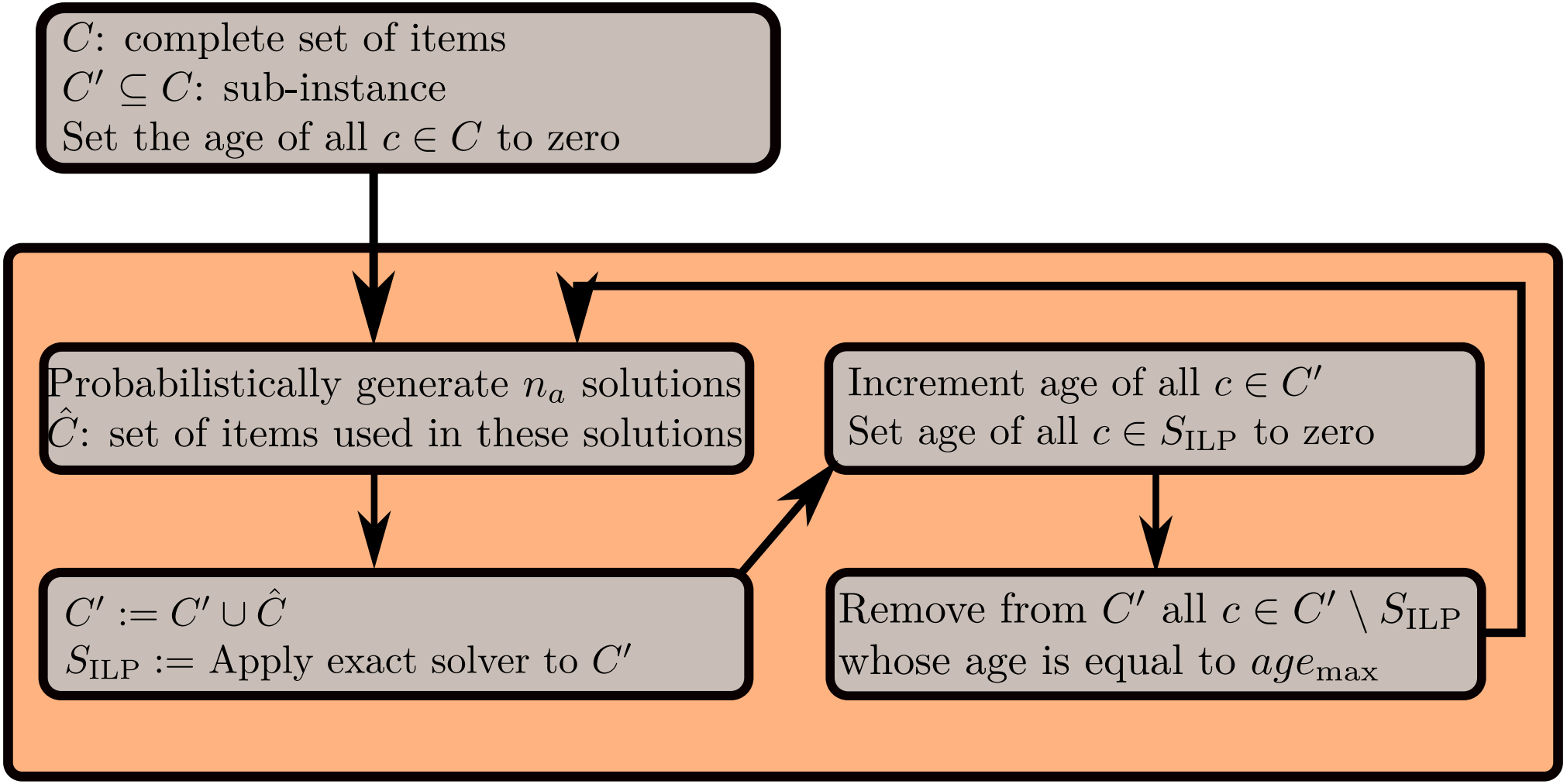
European Journal of Operational Research
Available online 13 August 2020
In Press, Corrected Proof

Discrete Optimization

A comparative analysis of two matheuristics by means of merged local optima networks

Christian Blum [a], Gabriela Ochoa [b]

Construct, Merge, Solve & Adapt: Flow Diagram



Test Case: Multi-dimensional Knapsack Problem (MDKP)

Given:

- A set of items $C = \{1, \ldots, n\}$
- A set of resources $K = \{1, \ldots, m\}$
- Of each resource k we have a maximum quantity c_k (capacity)
- Each item i requires from each resource k a certain quantity $r_{i,k}$
- Each item i has a profit p_i

Valid solutions: Each subset $S \subseteq C$ is a valid solution if

$$\sum_{i \in S} r_{i,k} \leq c_k \quad \forall k \in K$$

Objective function (to be maximized): $f(S) := \sum_{i \in S} p_i$ for all valid S

MDKP: Greedy Heuristic

Re-ordering of the items: with respect to utility values

$$u_i \leftarrow \frac{p_i}{\sum_{k \in K} r_{i,k}/c_k} \qquad i \in C.$$

Simple heuristic:

- Consider the list of re-ordered items of left to right
- Add each item that does not violate the resource constraints to the partial solution under construction

MDKP: ILP Model

Standard model:

$$\text{maximize} \quad \sum_{i \in C} p_i \cdot x_i \tag{1}$$

$$\text{subject to} \quad \sum_{i \in C} r_{i,k} \cdot x_i \leq c_k \qquad \forall k \in K \tag{2}$$

$$x_i \in \{0, 1\} \qquad \forall i \in C \tag{3}$$

Application to sub-instances:

- In LNS: fix all x_i to one $\forall\, i \in S_{\text{partial}}$.
- In CMSA: replace C by C'

MDKP: instance tightness

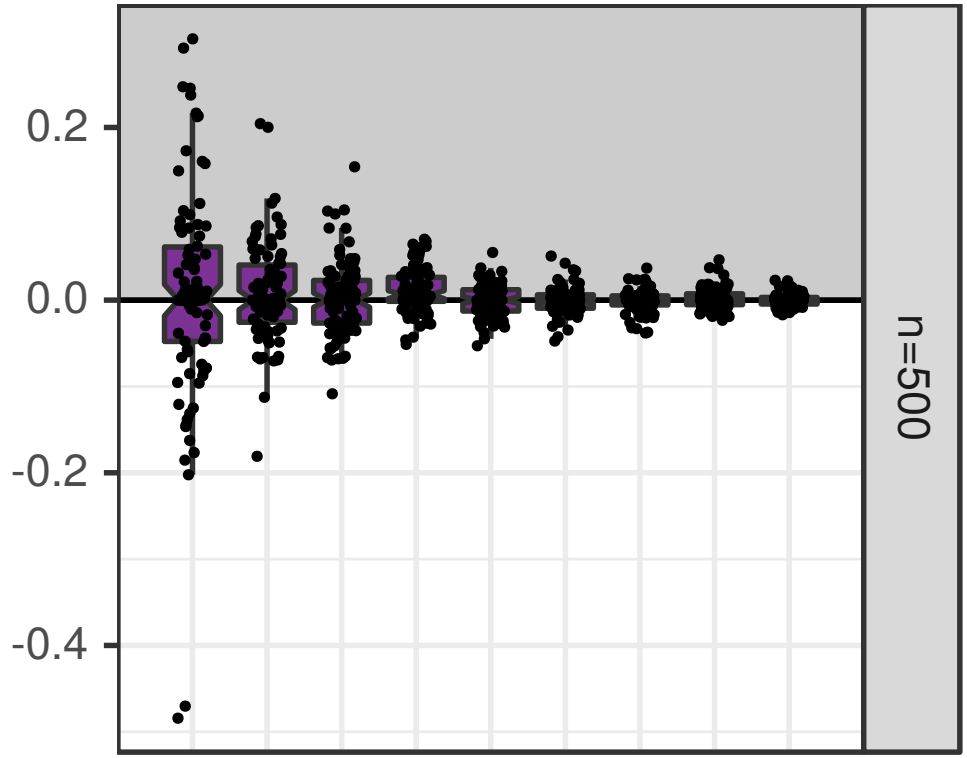
Important parameter: Instance tightness $0 \leq \alpha \leq 1$

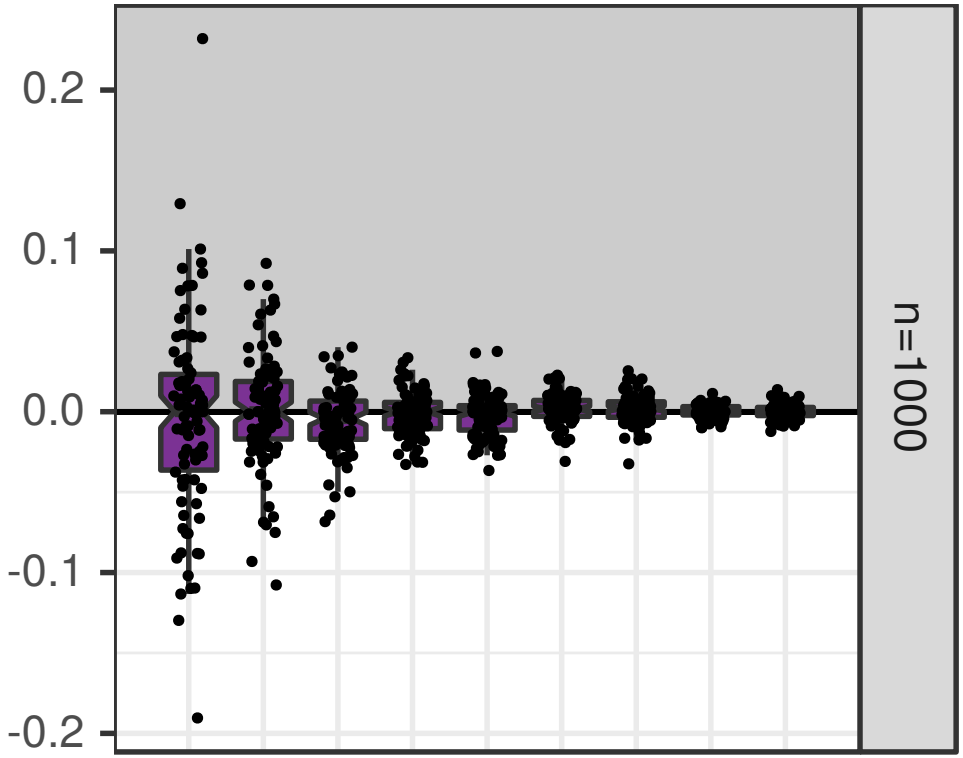
- When α close to zero: capacities are low and valid solution only contain very few items
- When α close to one: capacities are very high and solutions contain nearly all items

Plan:

- Apply both LNS and CMSA to instances from the whole tightness range.
- Both algorithms are tuned with irace seperately for instances of each considered tightness.

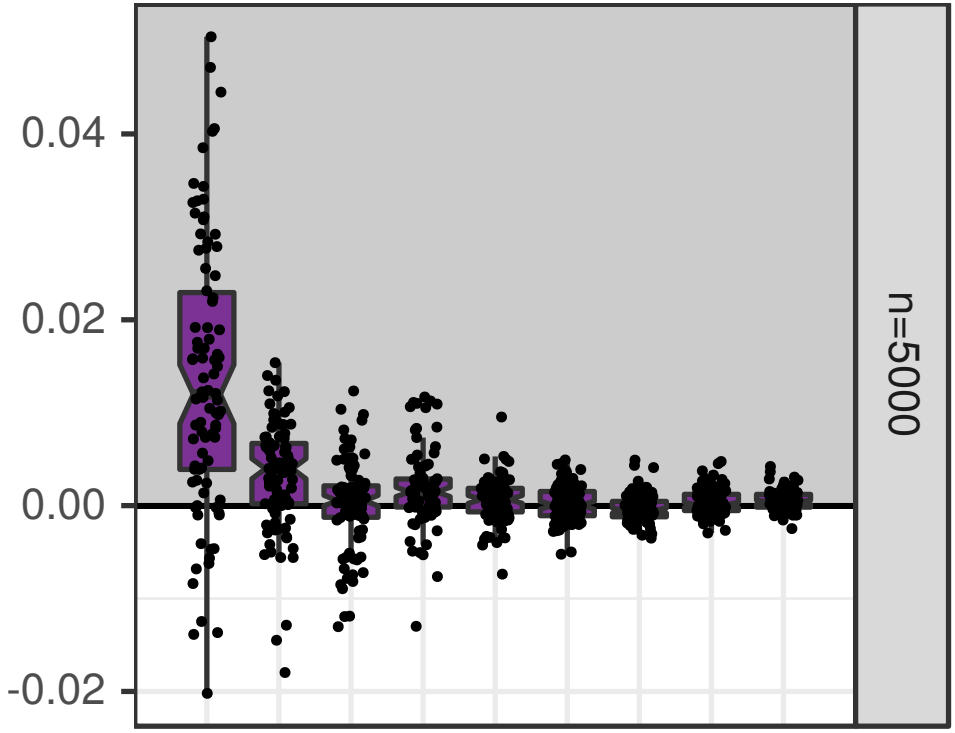
MDKP: LNS/CMSA comparison (1)

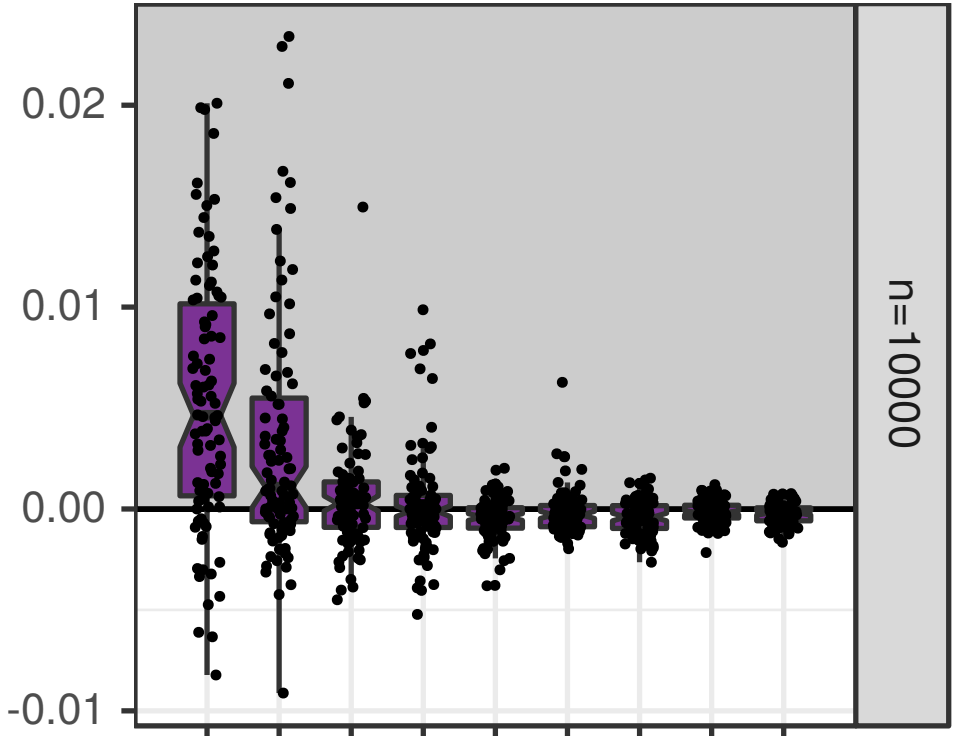




- X-axis: instances with increasing tightness (from left to right)
- Y-axis: improvement of CMSA over LNS (in percent)

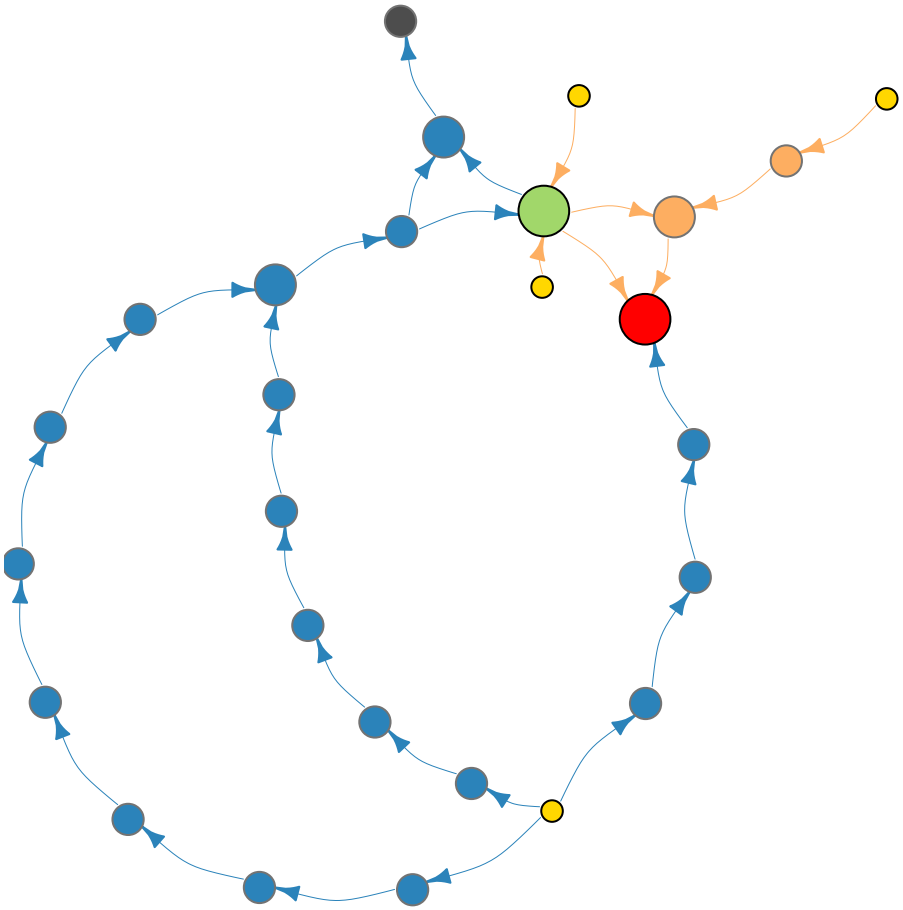
MDKP: LNS/CMSA comparison (2)





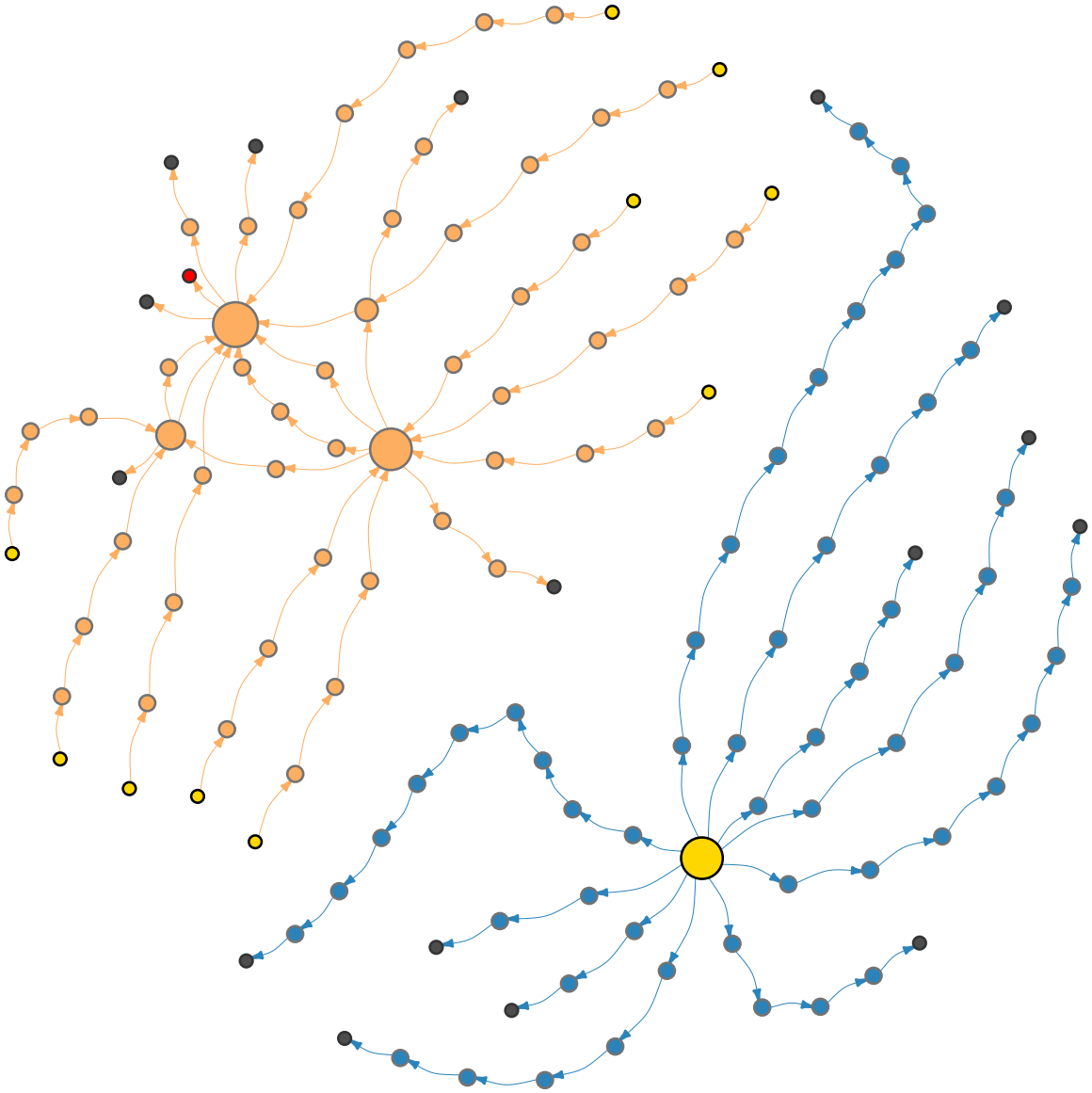
- X-axis: instances with increasing tightness (from left to right)
- Y-axis: improvement of CMSA over LNS (in percent)

MDKP: Comparative visualization of results

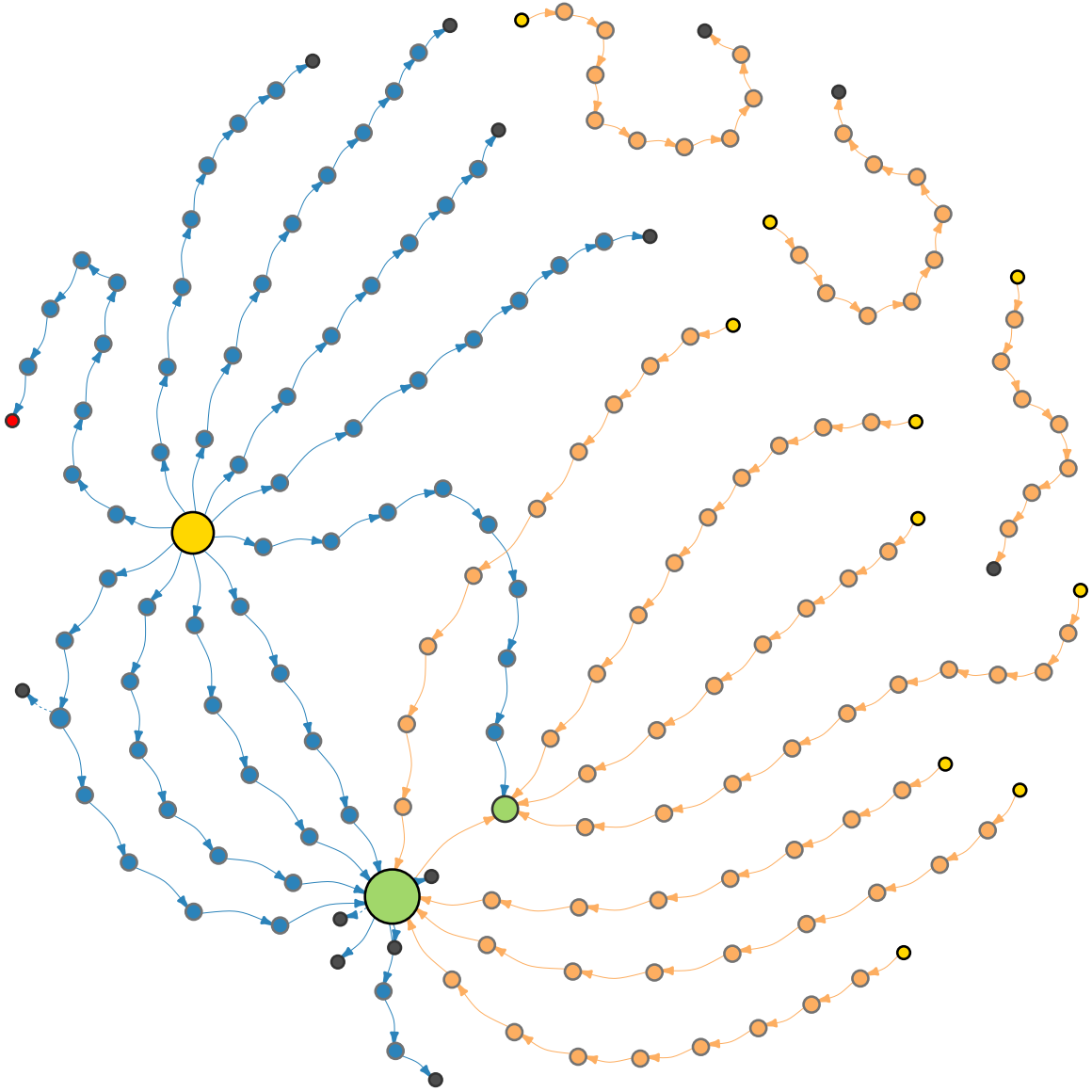


●	Solution visited by CMSA
→	Search transition of CMSA
●	Solution visited by LNS
→	Search transition of LNS
●	Solution visited by both algorithms
●	Solution at the start of a run (either algorithm)
●	Solution at the end of a run (either algorithm)
●	Best solution found across all runs and algorithms
Node Size	Proportional to incoming weighted degree

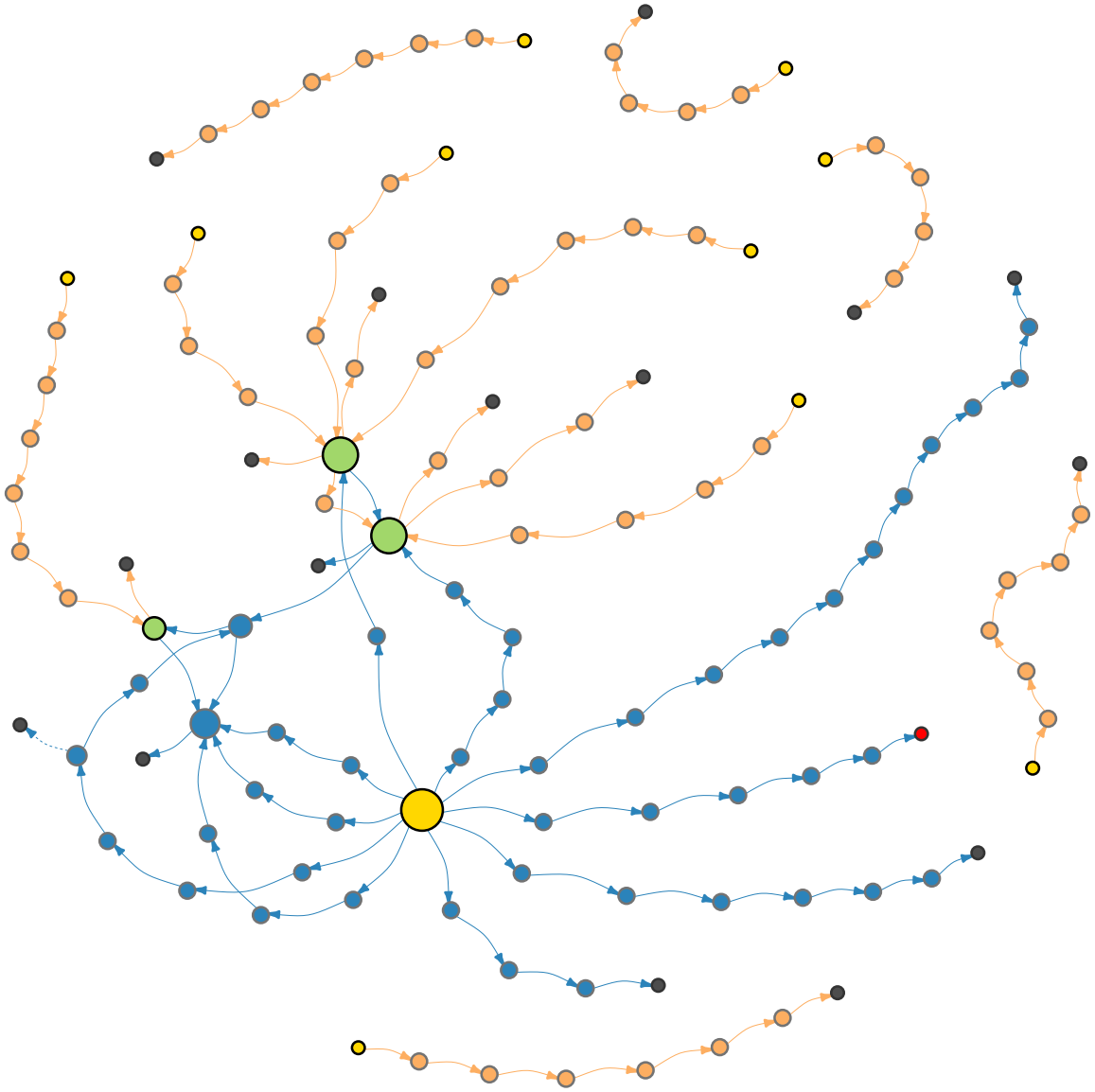
MDKP: Comp. visualization, lowest tightness ($n = 10000$)



MDKP: Comp. visualization, medium tightness ($n = 10000$)



MDKP: Comp. visualization, highest tightness ($n = 10000$)



And in comparison to the state of the art? (1)

ELSEVIER

Expert Systems with Applications
Volume 149, 1 July 2020, 113310

Diversity-preserving quantum particle swarm optimization for the multidimensional knapsack problem

Xiangjing Lai [a], Jin-Kao Hao [b, d], Zhang-Hua Fu [c, e], Dong Yue [a]

ELSEVIER

Information Sciences
Volumes 436–437, April 2018, Pages 282-301

A two-phase tabu-evolutionary algorithm for the 0–1 multidimensional knapsack problem

Xiangjing Lai [a], Jin-Kao Hao [b, c], Fred Glover [d], Zhipeng Lü [e]

And in comparison to the state of the art? (2)

- Benchmark: 30 instances with 500 items and 10 resources from the OR-Library (`http://people.brunel.ac.uk/~mastjjb/jeb/info.html`).
- Comp. setup : time limit 200 seconds, 100 runs per instance

Avg. solution quality:

TPTEA	DQPSO	CMSA	LNS
212840.70	212841.60	**212842.03**	212841.73

Avg. computation time:

TPTEA	DQPSO	CMSA	LNS
3467.04	120.26	**76.71**	108.06

Summary and Possible Research Directions

Summary: Hybrid metaheuristics

- **Reason of being:** especially due to the large scale of many real-life problems
- Advantage: they profit from the synergies obtained by combining different techniques
- Therefore: many optimization algorithms nowadays are hybrids

Possible Research Directions:

- Regarding existing approaches: contributing to their deeper understanding and their improvement
- Regarding new approaches: especially the combination with machine learning seems to offer opportunities

Thanks to colleagues involved in my research on hybrids

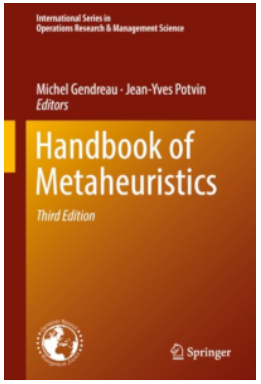
Pedro Pinacho

Günther R. Raidl

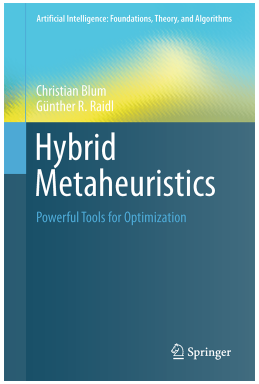
Gabriela Ochoa

Questions?

Recommended literature:



Edited Book: M. Gendreau, J.-Y. Potvin – Handbook of Metaheuristics, Springer Series on Operations Research & Management Science, 2019



Book: C. Blum, G. R. Raidl. Hybrid Metaheuristics – Powerful Tools for Optimization, Springer Series on Artificial Intelligence, 2016