Stealing items more efficiently with ants: a swarm intelligence approach to the travelling thief problem

THE UNIVERSITY of ADELAIDE

Markus Wagner
markus.wagner@adelaide.edu.au
http://cs.adelaide.edu.au/~markus

Background

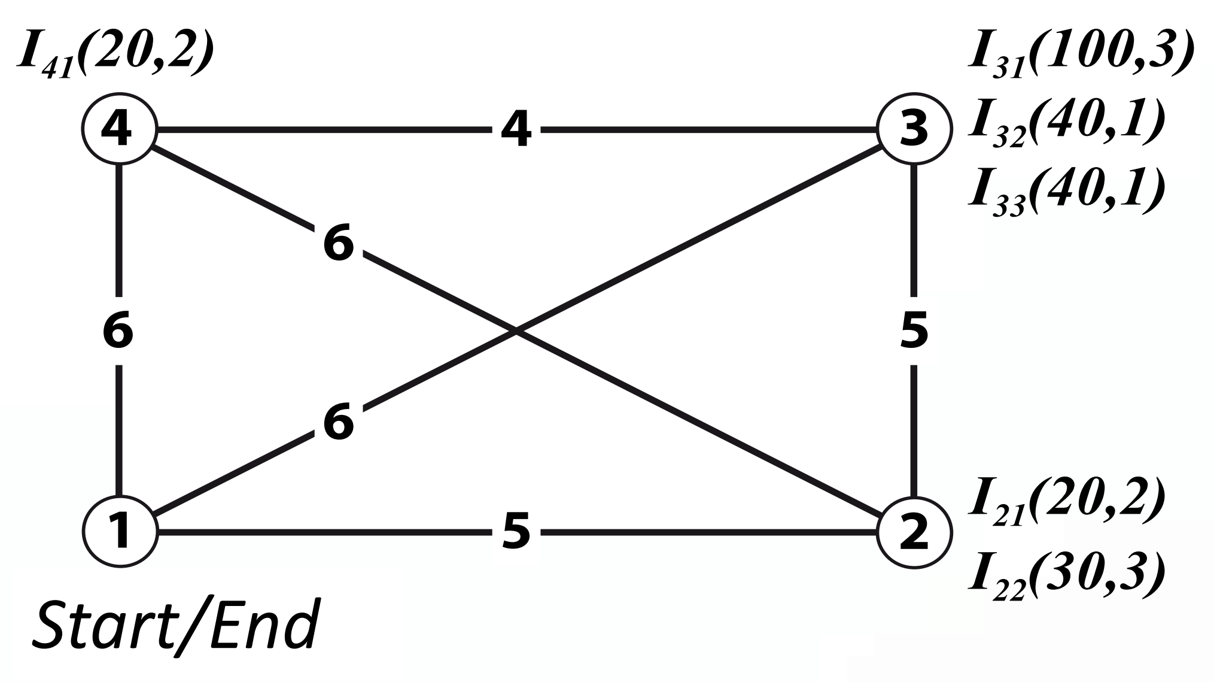
Many real-world problems are composed of several interacting components. In order to facilitate research on such interactions, the Traveling Thief Problem (TTP) was created in 2013 as the combination of two well-understood combinatorial optimization problems, which are the traveling salesperson problem (TSP) and the knapsack problem (KP).

TTP formulation in a nutshell:

objective score = profit of items – renting rate * travel time

where the travel time is dependent on the load, which results in **interdependent sub-problems**. Consequently, the optimal TTP solution does typically not make use of the optimal TSP or KP solution.

Illustrative example for a TTP instance with four cities and six items:



Our Approach

- **Let ants do the TSP part, without telling them about the TTP!**

- Use FastPacking (from literature) to solve the KP part for each ant-generated tour, resulting in a TTP objective score for each ant-generated tour. This step can be seen as some form of **bi-level optimisation**.

➔ Ant scores are these TTP objective scores, not the raw TSP tour lengths.

Technical details

- Minimal customisation of **ACOTSP**, no parameter tuning.
- Caching of *<tour,objective score>* tuples to **reduce runtime**.
- Rotation of tours was necessary to start at City 1.

Pseudo code

```
Algorithm 1 ACOTSP for the Travelling Thief Problem (injections in italics)
1: while (termination condition not met)
2:     Construct tours using ants.
3:     Construct for each tour a packing plan using PackIterative, resulting in a
       TTP objective score. If the tour has been assessed before, we skip the packing
       step and retrieve the score from a cache.
4:     Perform local search on tours (if activated).
5:     Update ACO statistics.
6:     Boost solutions using (1+1)-EA, Insertion, Bitflip (if activated).
7:     Pheromone trail update.
```

Existing Approaches

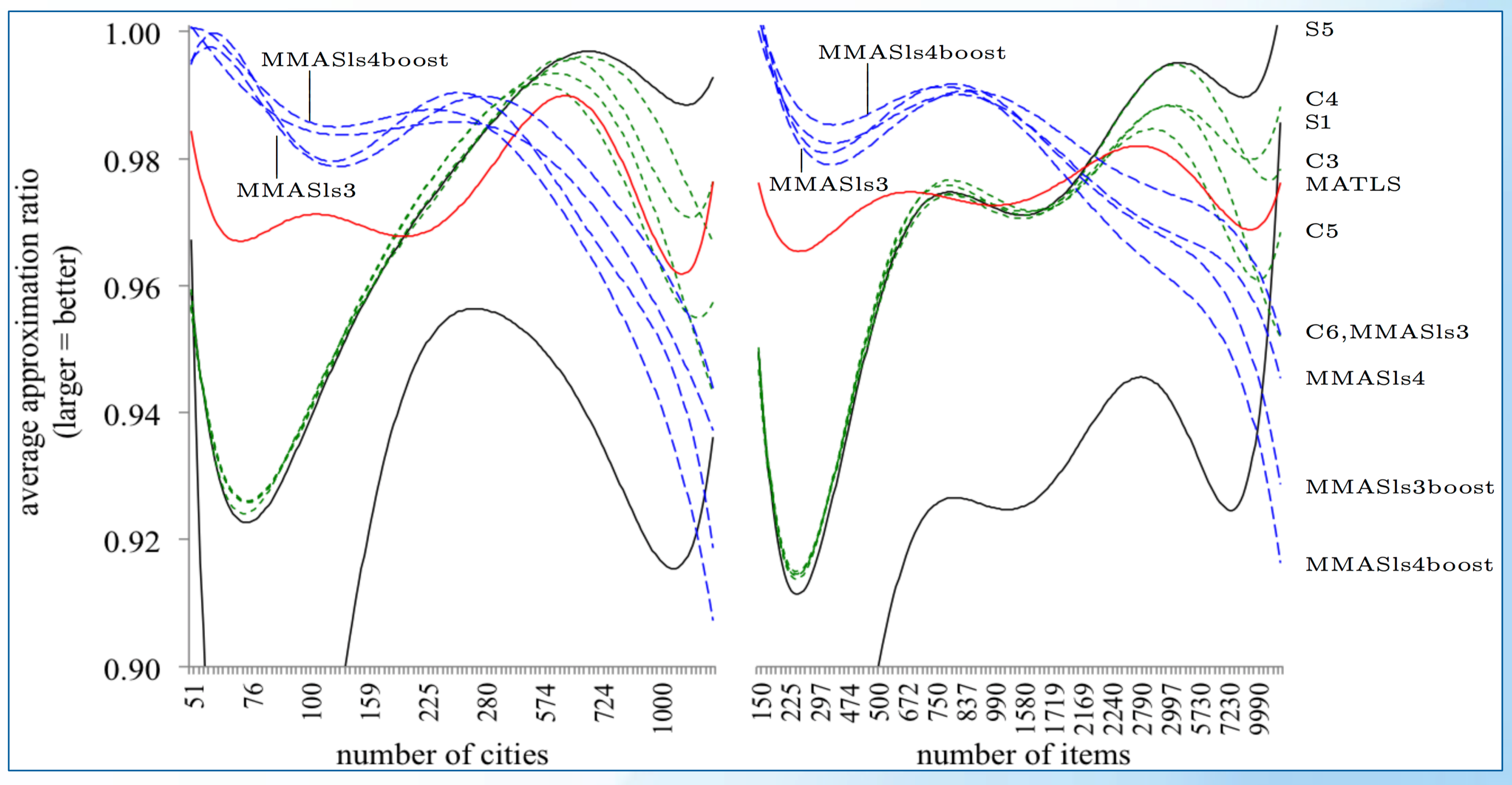
- Often: given a near-optimal tour, construct a picking plan.
- Sometimes: additional hill-climbers added that attempt tour changes.

➔ significant **bias towards very short tours** *(unjustified?)*

Other approaches

- Co-evolution: more holistic, but weak results so far.
- Constructive heuristics: fast, but missing the hill-climber.
- Variation operators are typically not TTP-specific, but only TSP- or KP-specific.

Results



Summary of results across 108 instances shown as trend lines (polynomials of degree six).

- Similar approaches are coloured identically: S1/S5, C3-C6, MATLS, MMAS.
- Our MMAS-based approaches are the best performing ones for TTP instances with up to 250 cities and 2000 items, on which previously MATLS and C3-C6 performed best.

approach	used knapsack capacity	unused knapsack capacity	total profit of items	travel distance	travel time	objective score	average approximation ratio
			eil51_n150_uncorr_07				
MMASls4	36538	11671	53368	467.00	652.11	11763	0.997
S1/C6	34622	13587	52145	459.00	659.35	10079	0.856/0.857
			dsj1000_n2997_uncorr-similar-weights_03				
MMASls4	758385	62635	590594	19286106	27205708	46480	0.832
MMASls3boost	774523	46497	595519	19290271	26699765	61524	0.871
S1	758408	62612	584276	18705228	26709155	50093	0.622
S5	758364	62656	590515	18750512	26599551	58524	0.931
C6	761602	59418	587164	18750975	26376923	59626	0.876

Details of best solutions (w.r.t. objective score) found in 30 runs, for two instances. Shaded are travel distances, best objectives scores and best approximation ratios.

- **MMAS tours of best solutions are the longest**.
- MMASls3boost found an **outstanding solution** for the second instance, however, it is outperformed on average by S5 (0.871 vs 0.931).

What's next???
Enough heuristics… how about understanding the TTP!!!

Online resources

- **http://tinyurl.com/ttpadelaide**
- code (this one and from other TTP projects), all 9720 instances, lots of results, …