An Ant Colony Optimization Approach to the Traveling Tournament Problem

David C. Uthus
Patricia J. Riddle
Hans W. Guesgen

Traveling Tournament Problem

- Sports scheduling combinatorial optimization problem.
- Objective is to create a double round robin tournament with minimal travel distance.

Round / Team	1	2	3	4	5	6
A	@B	@C	@D	B	C	D
B	A	D	@C	@A	@D	C
C	@D	A	B	D	@A	@B
D	C	@B	A	@C	B	@A

Traveling Tournament Problem

- TTP takes in *n* (even) teams and distance matrix.
- Double round robin requires each team to play every other team twice, once home and once away.
- Each team must play once every round.

Round / Team	1	2	3	4	5	6
A	@B	@C	@D	B	C	D
B	A	D	@C	@A	@D	C
C	@D	A	B	D	@A	@B
D	C	@B	A	@C	B	@A

Traveling Tournament Problem

- *At_most* constraint restricts number of consecutive home and away games to 3.
- *No_repeat* constraint prevents any team from playing another team consecutive rounds.

Round / Team	1	2	3	4	5	6
A	@B	@C	@D	B	C	D
B	A	D	@C	@A	@D	C
C	@D	A	B	D	@A	@B
D	C	@B	A	@C	B	@A

Traveling Tournament Problem

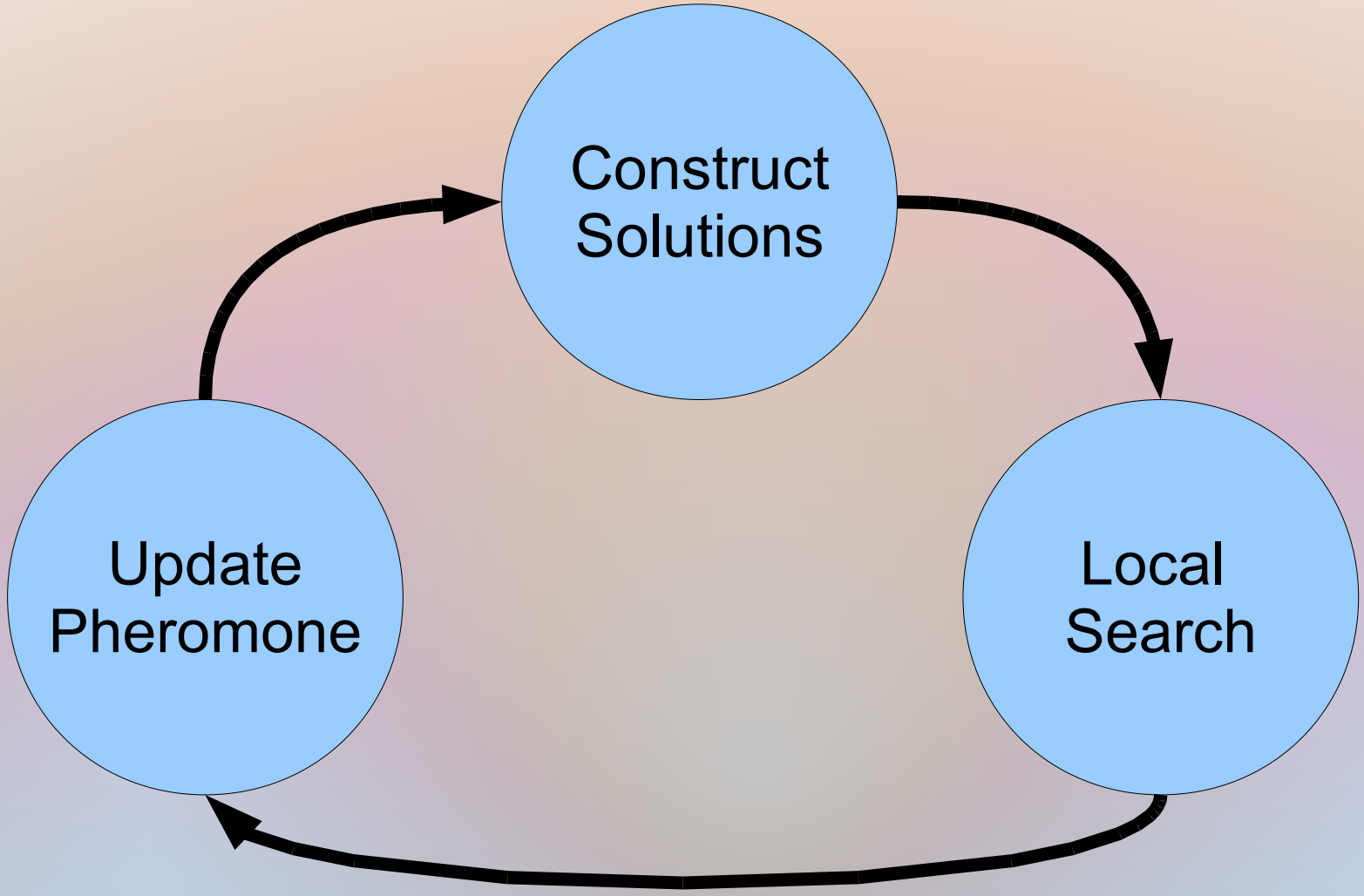
- Distances calculated individually for each team, similar to Traveling Salesman Problem
- Objective is to minimize total summed distance of all teams.

Round / Team	1	2	3	4	5	6
A	@B	@C	@D	B	C	D
B	A	D	@C	@A	@D	C
C	@D	A	B	D	@A	@B
D	C	@B	A	@C	B	@A

Traveling Tournament Problem

- Related to real world problem of scheduling Major League Baseball.
- Difficult problem to solve to optimality, only smallest instances have been solved.
- Most best solutions have been found by metaheuristics.
- Many were found with Population – Based Simulated Annealing using 80 cores.

Ant Colony Optimization



Ant Colony Optimization

- ACO has had very poor results when applied to TTP compared with other metaheuristic approaches.
- Crauwels and Van Oudheusden (2003) were first to apply with direct approach.
- Chen et al. (2007) were second to use ACO, used it as hyper-heuristic.

Ant Colony Optimization

- Problem of applying ACO to TTP lies in constructing solutions.
- Problem contains hard constraints, need to use backtracking.
- Using only backtracking can result in solutions taking too long to construct past 10 teams.

FC-CBJ

- Integrate ACO with FC-CBJ.
- Allows forward checking of constraints along with backjumping.
- Easy to integrate, only have to change way values are chosen.
- Further improve hybridization by using unsafe backjumping and ant restarts.

Unsafe Backjumping

- Safe backjumping ensures no feasible solutions will be missed while constructing solutions.
- ACO is a probabilistic approach, don't have to be concerned with safe backjumping.
- Unsafe backjumping allows the ant to get out of constructing an infeasible solution faster.

Ant Restarts

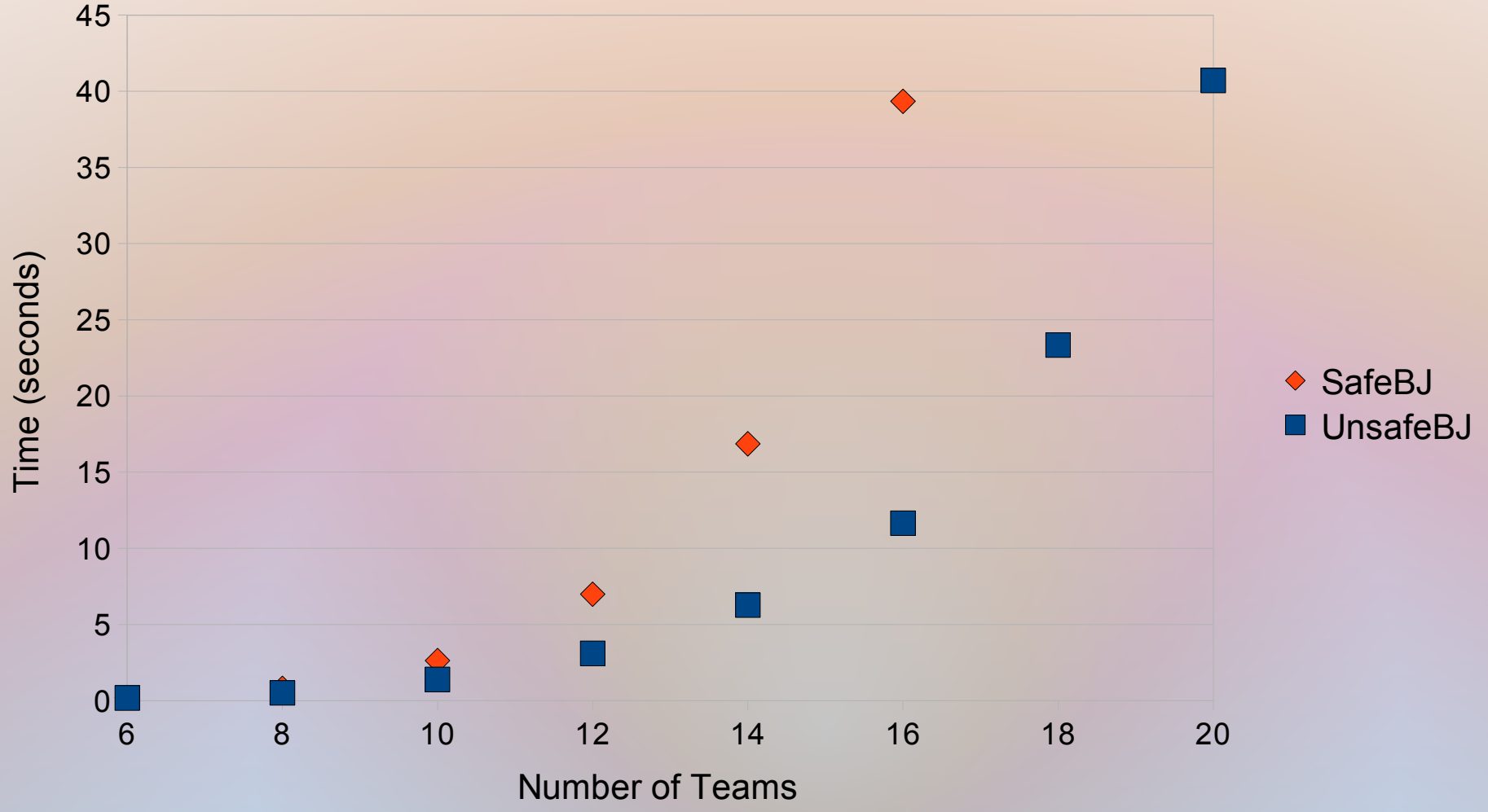
- Restart an ant after it has performed a certain number of backjumps.
- Helps to handle rare cases where we can't propagate constraints.
- Differs from past approaches in that it has tolerance for some backjumping/backtracking before restarting.

Constructing Solutions

- Construct solutions from rounds 1 to *r*. Assign all teams for a round before starting next round.
- Easy to propagate constraints.

Round / Team	1	2	3	4	5	6
A	@B	@C				
B	A					
C	@D	A				
D	C					

Constructing Solutions



Pattern Matching

- Use a new idea of pattern matching for constraint propagation, specifically for the *at_most* constraint.
- Create patterns at start of running application, take very little time even with 32 teams.
- Helps to greatly reduce the amount of backjumping and ant restarts needed.

Pattern Matching

- Find patterns in number of remaining home and away games with regards to *at_most* constraint:

  1H, 6A : *AAAHAAA*

  1H, 5A : *AABBAA*

  1H, 4A : *ABBBA*

  1H, 3A : *BBBB*

Pattern Matching

1H, 4A : *ABBBA*

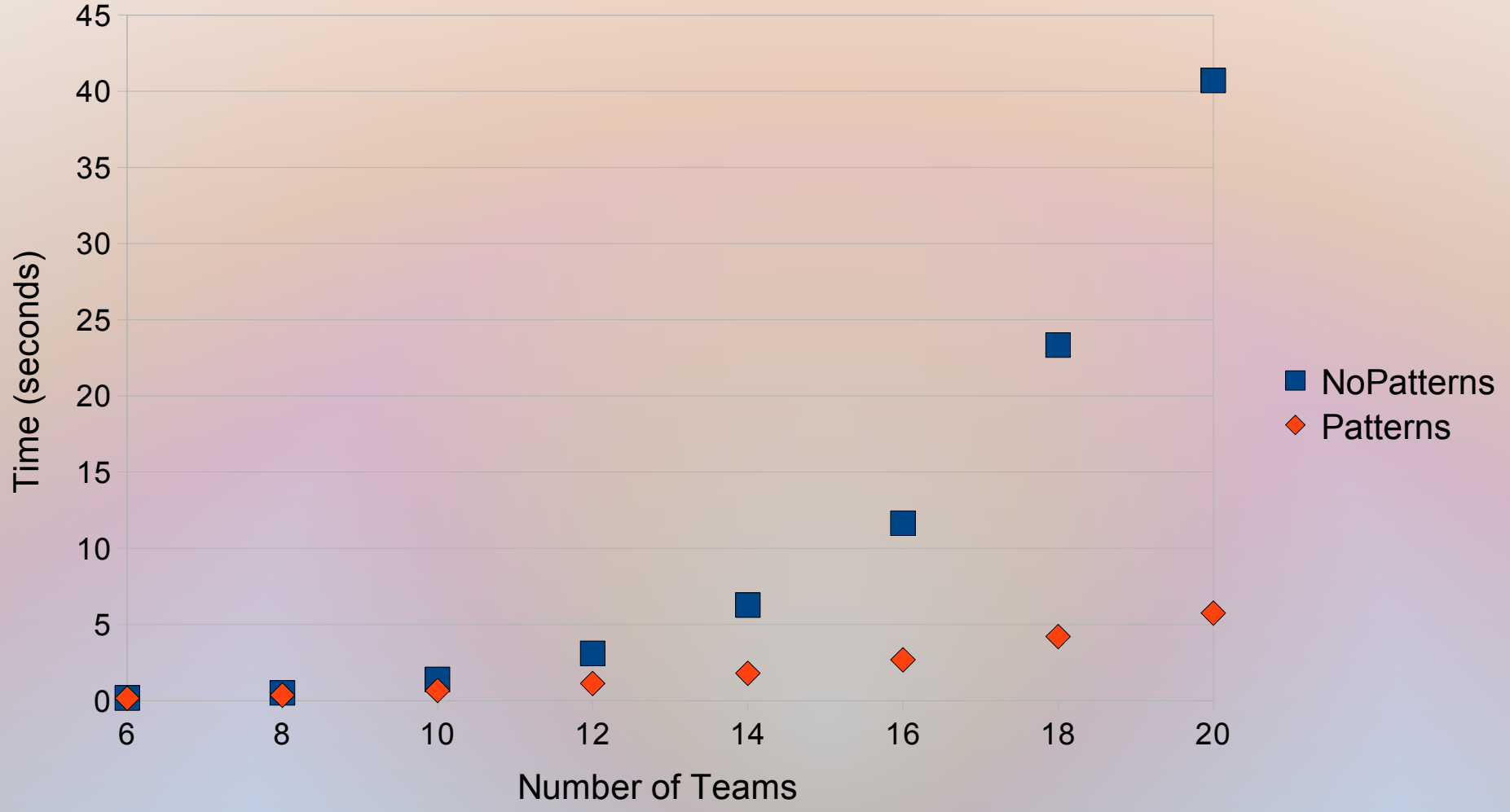
H	1		1		1
A	2 4	2	5	2 3	

Pattern Matching

1H, 4A : *ABBBA*

H	~~1~~		1		~~1~~
A	2 4	2	5	2 3	

Constructing Solutions



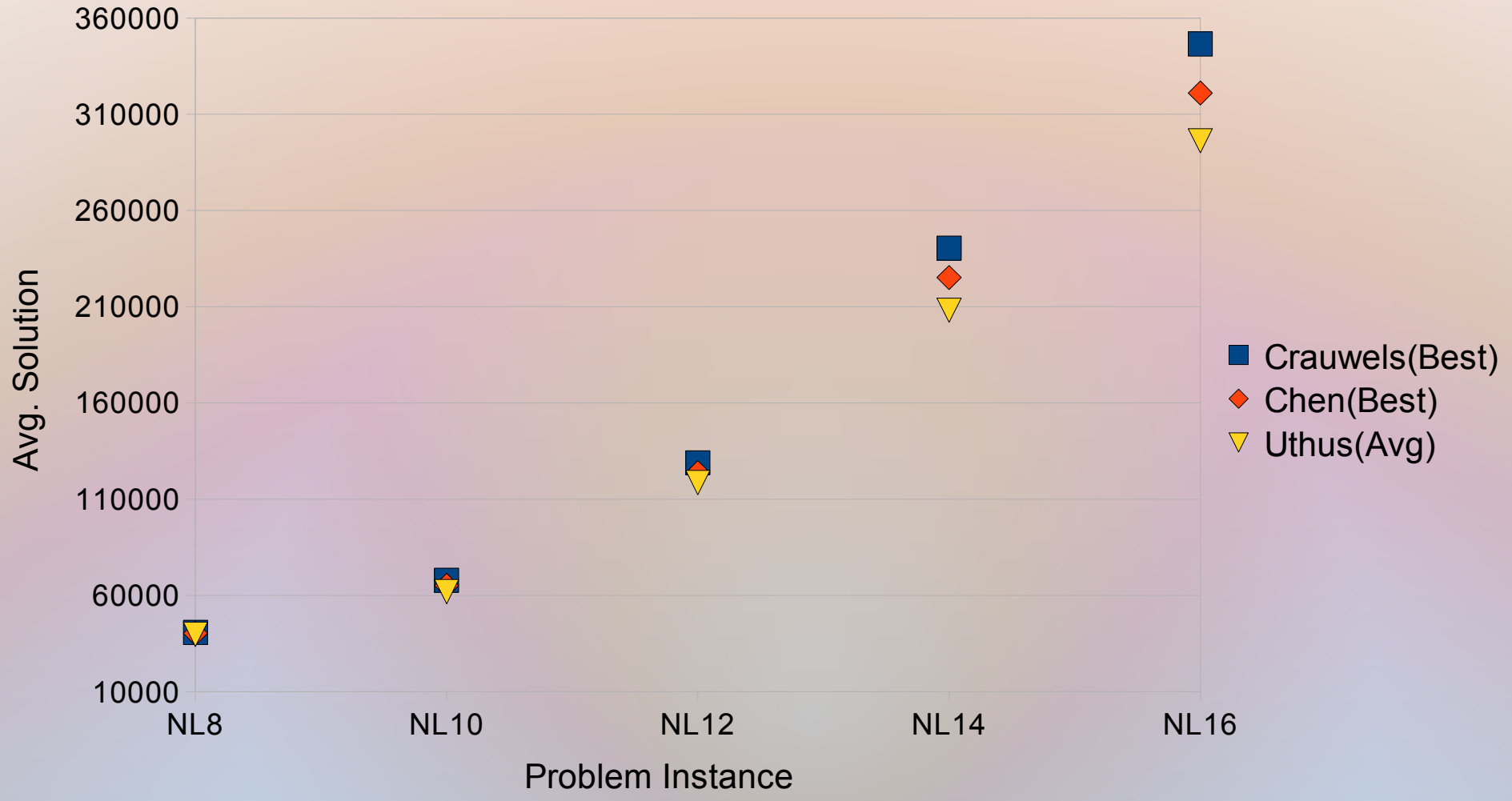
Pheromone

- Represents the desirability of having team *i* play at team *j* during round *r*.
- Updated at end of each cycle using the best ant of either the current iteration or best seen since last pheromone update.
- Use pheromone restarts to help prevent stagnation.

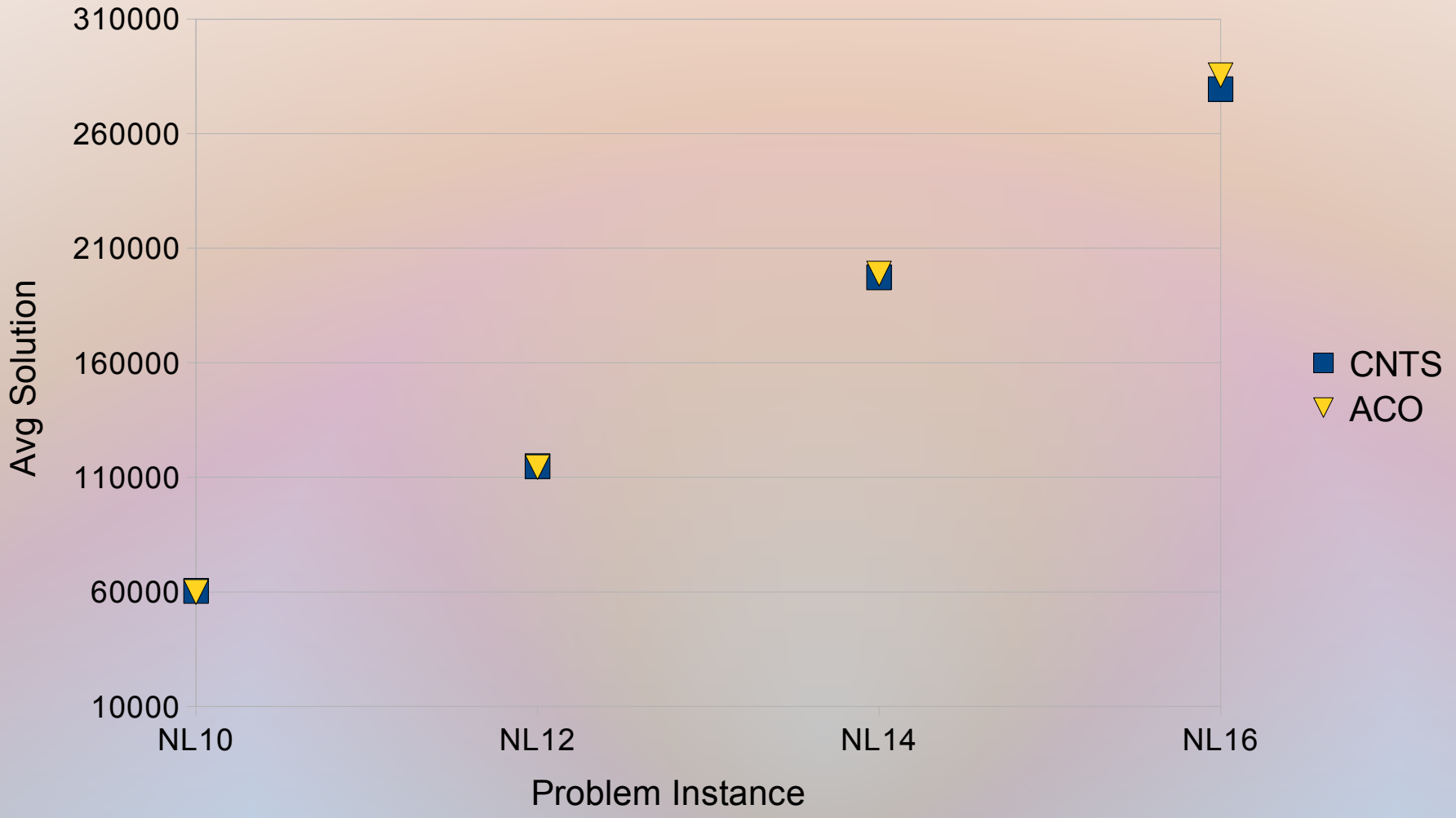
Local Search

- Use a tabu search approach for the local search.
- Use same neighborhood search as simulated annealing.
- Applied to all ants at end of solution construction phase.

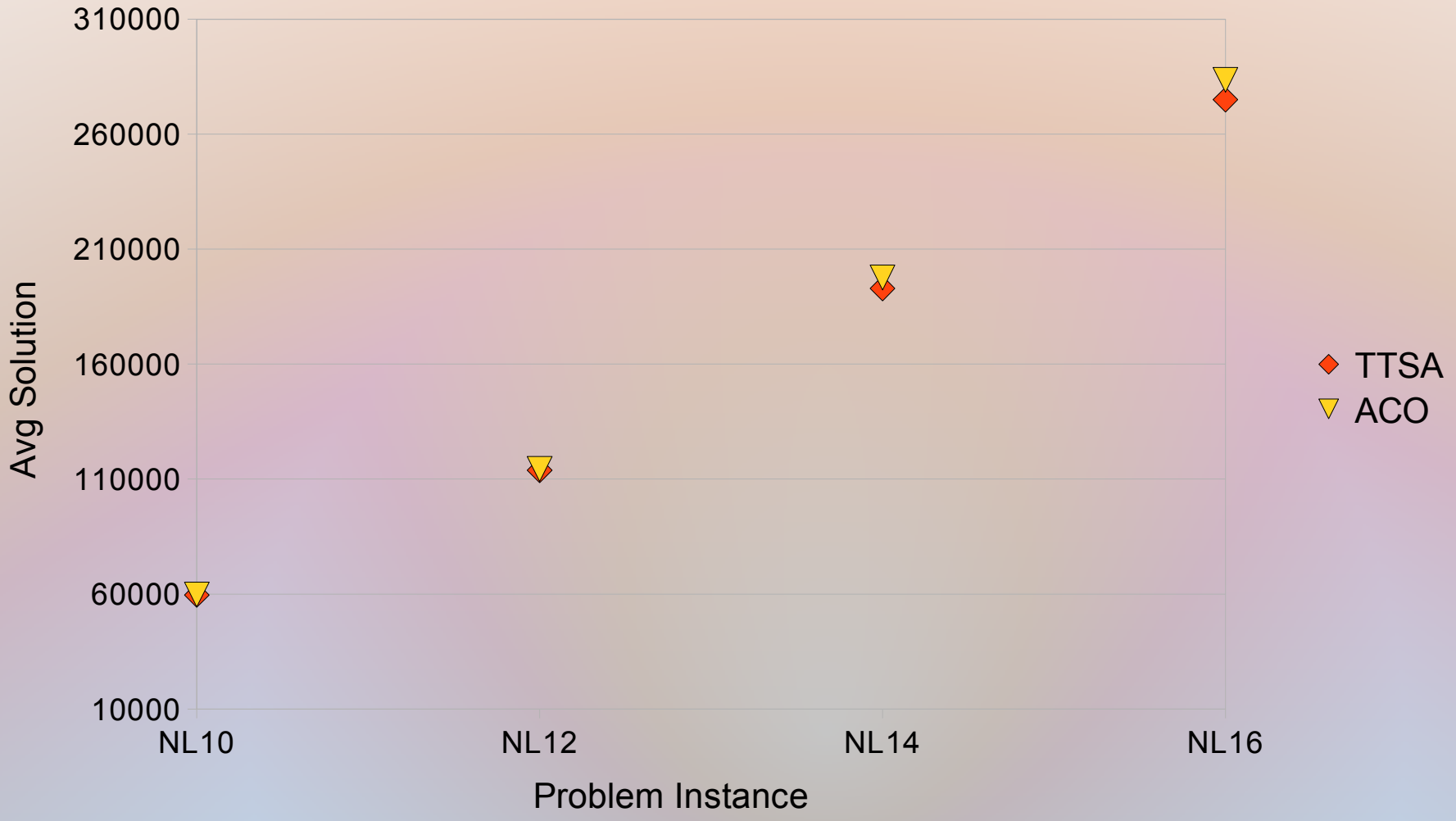
Results


360000
310000
260000
210000
160000
110000
60000
10000
Avg. Solution
NL8
NL10
NL12
NL14
NL16
Problem Instance
Crauwels(Best)
Chen(Best)
Uthus(Avg)

Results


310000
260000
210000
160000
110000
60000
10000
Avg Solution
NL10
NL12
NL14
NL16
Problem Instance
CNTS
ACO

Results


310000
260000
210000
160000
110000
60000
10000
Avg Solution
NL10
NL12
NL14
NL16
Problem Instance
TTSA
ACO

Future

- Look at different definitions of pheromone matrix.
- Look into decreasing time needed to find good solutions, even if cant beat best solutions.
- Optimizing usage of local search, since local search is very time-consuming compared to constructing solutions.

Conclusions

- New integration of ACO with FC-CBJ.
- New idea of using pattern matching for constraint propagation.
- Greatly improved performance of applying ACO to the TTP.
- Results comparable to state-of-the-art approaches.

Thank You