ITSA*: Iterative Tunneling Search with A*

David A. Furcy
(University of Wisconsin)

Action Elimination and Plan Neighborhood Graph Search: Two Algorithms for Plan Improvement

Hootan Nakhost, Martin Müller
(University of Alberta)

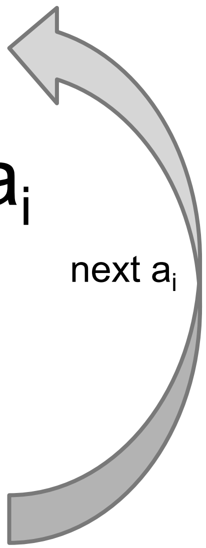
Basil Kohler, 1.11.2012

Topic

- Improving suboptimal solutions
  - Take a solution
  - (Iteratively) improve solution
- Anytime algorithms - run until
  - There is no time left
  - There is no memory left
  - The solution cannot be improved by the algorithm

Action Elimination (AE)

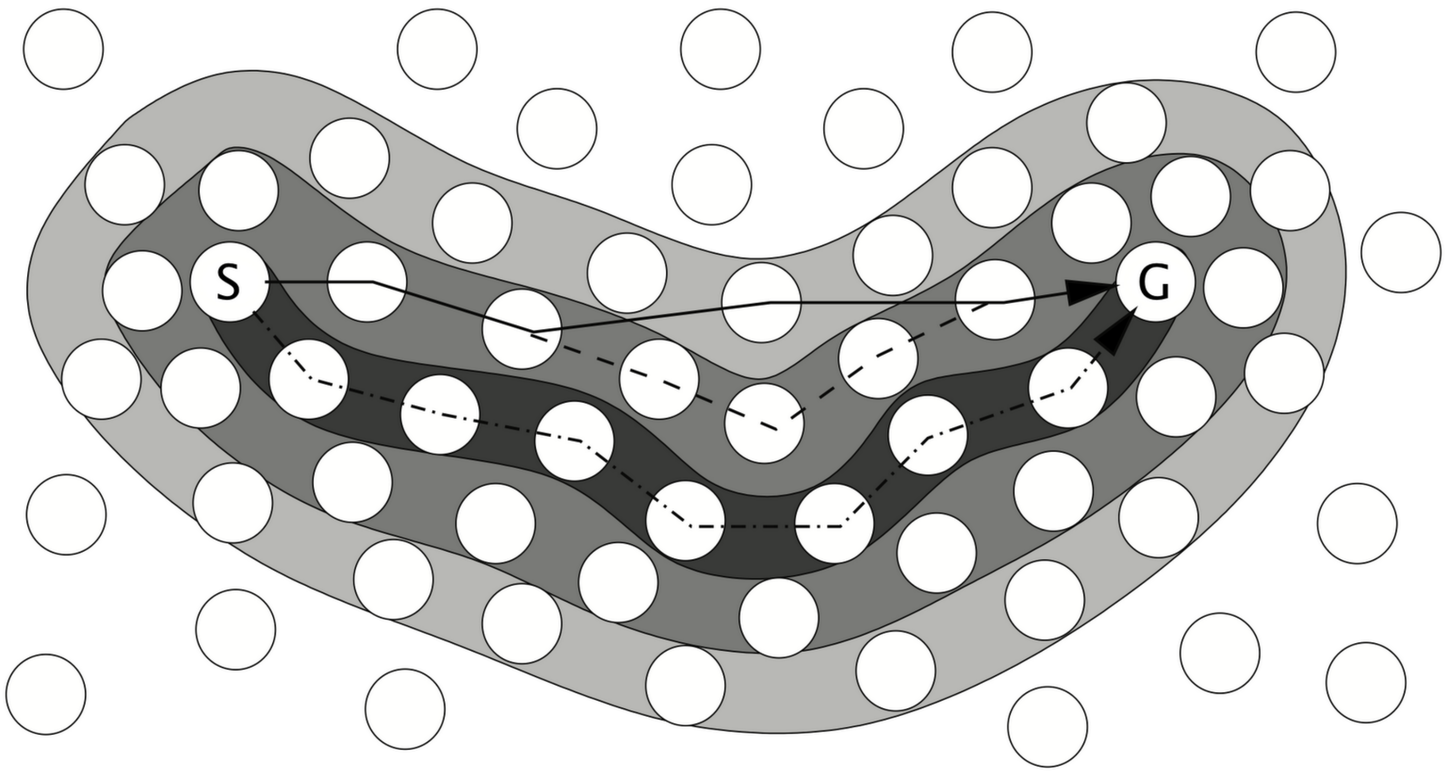
- STRIPS planning task with suboptimal plan $\pi = (a_1, a_2, ..., a_n)$
- Remove a_1
- Try a_2 ... a_n, remove each non-applicable a_i
- π not valid:
  - re-add removed actions to π
- Continue testing next actions analogously



AE Properties

- Works directly on the plan
- Shorter π better → interrupt any time
- Simple and fast
- Greedy
- $O(p\ n^2)$
  - p: max. number of preconditions
  - n: length of plan

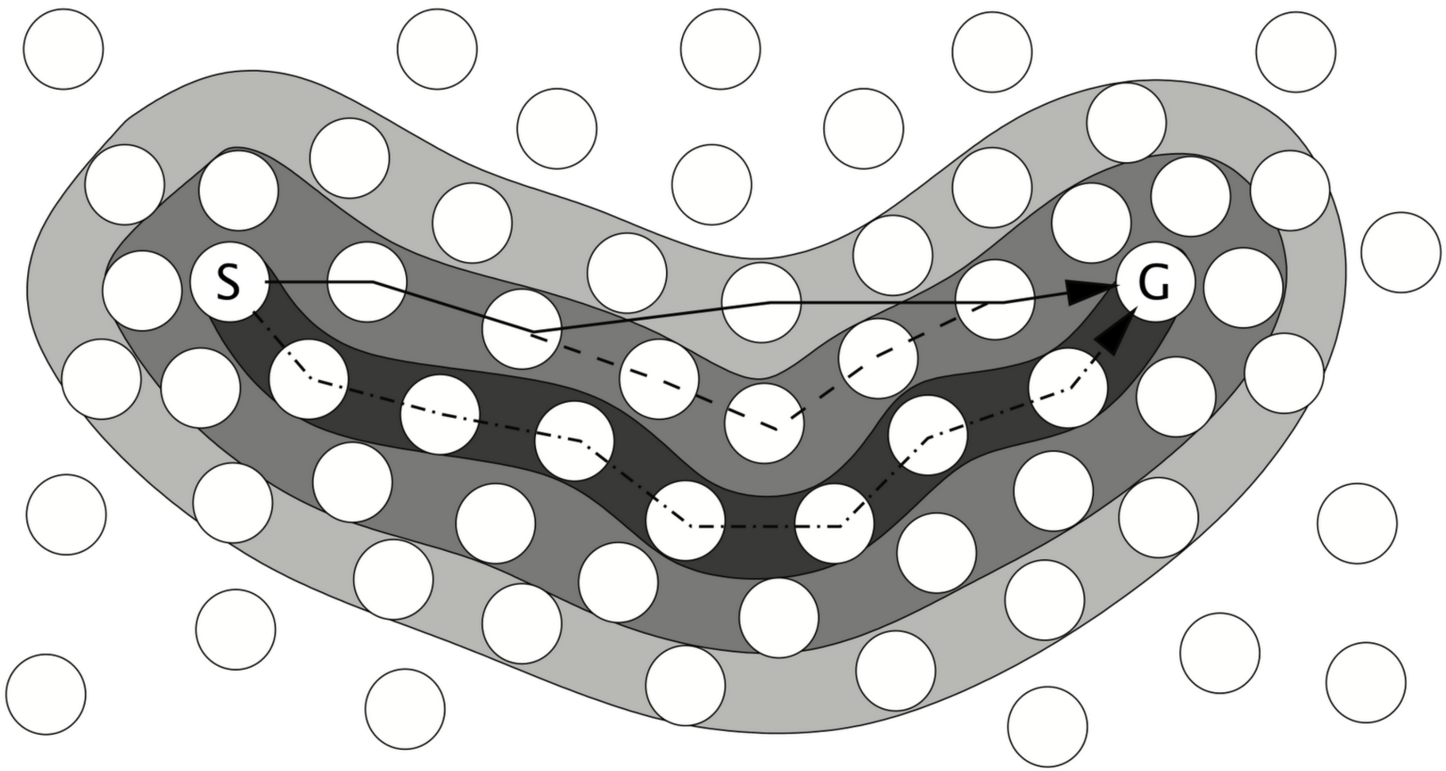
Illustration



ITSA*: It's a star!
Iterative Tunneling Search with A*

- Solution path P
- States in P get a value of 0, childs 1
- Modified A* search from start s_0 with iteration number (IN) 1:
  - Only add states to OPEN if current IN ≥ $value_{state}$
  - Created states get parents value +1
  - Stop when the goal is found
- Restart A* search from s_0 with incremented iteration number

Illustration



One-step and Multi-step ITSA*

- One-step ITSA*
  - Run ITSA* until memory is full
- Multi-step ITSA*
  - Run ITSA* until memory is full
  - Take new solution, re-run ITSA* until memory is full
  - Repeat until:
    - no time left
    - the solution does not change

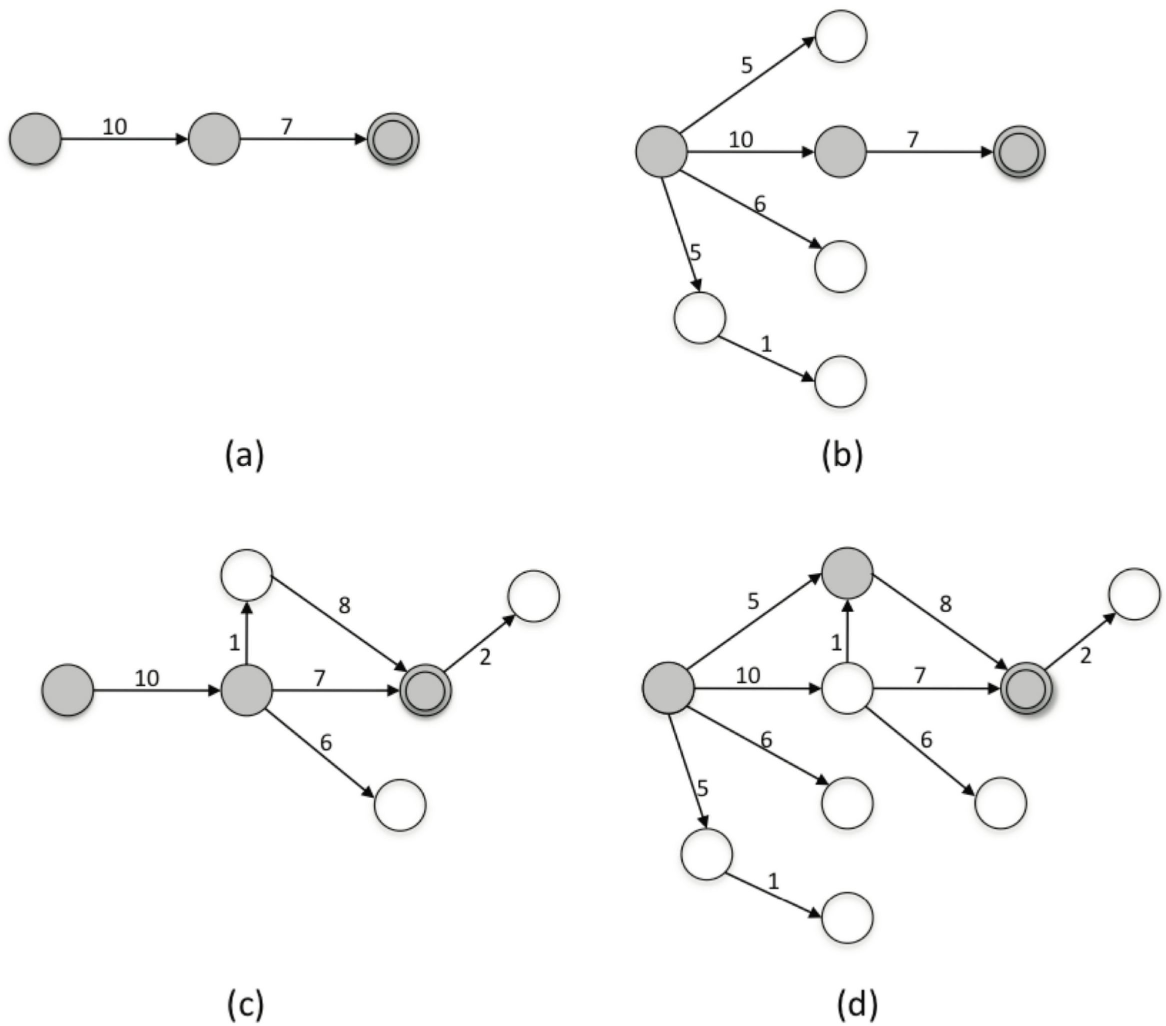
ITSA* Properties

- Anytime
  - Each ITSA* iteration can improve solution
  - Each ITSA* step can improve solution
- No parameter
- Optimal plan given enough time and memory
  - last iteration: A* search with all states

Neighborhood Graph Search (NGS)

- Set of states of current solution
- M searches each state with exploration limit L
  - M: deterministic graph search method
  - L: limit on number of expanded nodes
- NG: Subgraph explored by these searches
- Compute shortest path from s_0 to G in NG
  - A*, Dijkstra ...

Searching the Neighborhood



NG Search Properties

- Mix of optimal and greedy search:
  - Greedy: Build neighborhood with limit L
  - Optimal: Shortest path in neighborhood
- Anytime
  - Start with small L
  - New solution as input, double L
  - M-search is bound by $(L+1)(n+1)$ states
- Build NG with different M's (e.g., A* and bbfs) and combine them
  ⇒ more "general" neighborhood?
- Parameter L

One-step ITSA*

B	BULB		BULB + one-step ITSA*					
	time (seconds)	cost	time (seconds)			cost		
			value	increase over BULB		value	decrease over BULB	
				absolute	relative		absolute	relative
5	0.1	11,737	5.7	5.6	5,600%	3,140	8,597	73%
10	0.9	36,282	6.7	5.8	644%	3,233	33,049	91%
100	6.1	14,354	12.2	6.1	100%	2,052	12,302	86%
1,000	7.3	1,409	12.8	5.5	75%	746	663	47%
10,000	21.7	440	27.7	6.0	28%	428	12	3%

Table 1: Performance of one-step ITSA* on paths found by BULB in the 48-Puzzle (with 6 million nodes in memory)

B	BULB		BULB + one-step ITSA*					
	time (seconds)	cost	time (seconds)			cost		
			value	increase over BULB		value	decrease over BULB	
				absolute	relative		absolute	relative
10	96.9	108,804.8	100.7	3.8	4%	94,346.6	14,458.2	13%
100	5.1	1,893.9	7.9	2.8	56%	679.0	1,214.9	64%
1,000	7.4	275.8	10.2	2.8	38%	178.5	97.3	35%
10,000	13.8	53.6	18.5	4.7	34%	47.3	6.3	12%
50,000	39.2	31.2	46.0	6.8	17%	30.6	0.6	2%
70,000	51.1	30.0	57.3	6.2	12%	28.7	1.3	4%
100,000	74.8	28.1	81.3	6.5	9%	27.6	0.5	2%
120,000	127.2	26.0	134.8	7.6	6%	25.7	0.3	1%

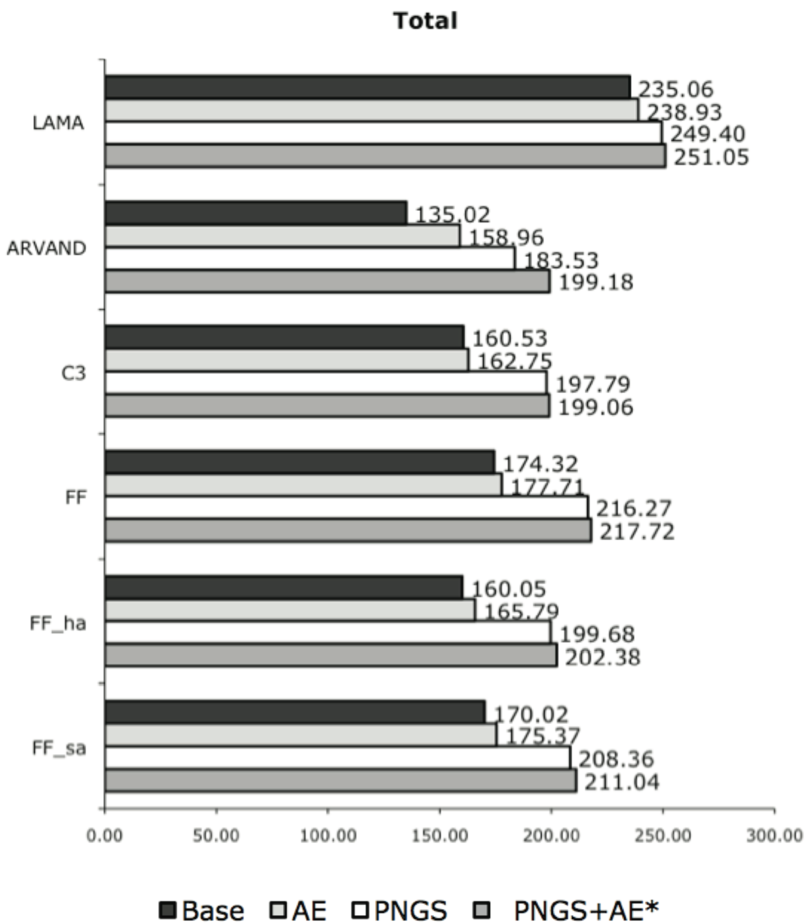
Table 2: Performance of one-step ITSA* on paths found by BULB in the Rubik's Cube (with 3 million nodes in memory)

Multi-step ITSA*

B	BULB		BULB + one-step ITSA*					
	time (seconds)	cost	time (seconds)			cost		
			value	increase over BULB		value	decrease over BULB	
				absolute	relative		absolute	relative
5	0.1	11,737	5.7	5.6	5,600%	3,140	8,597	73%
10	0.9	36,282	6.7	5.8	644%	3,233	33,049	91%
100	6.1	14,354	12.2	6.1	100%	2,052	12,302	86%
1,000	7.3	1,409	12.8	5.5	75%	746	663	47%
10,000	21.7	440	27.7	6.0	28%	428	12	3%

B	BULB		BULB + multi-step ITSA*					
	time (seconds)	cost	time (seconds)			cost		
			value	increase over BULB		value	decrease over BULB	
				absolute	relative		absolute	relative
5	0.1	11,737	50.3	50.2	50,200%	2,562	9,175	78%
10	0.9	36,282	53.0	52.1	5,789%	1,808	34,474	95%
100	6.1	14,354	55.2	49.1	805%	1,159	13,195	92%
1,000	7.3	1,409	36.7	29.4	409%	674	735	52%
10,000	21.7	440	42.4	20.7	95%	426	14	3%

AE, PNGS, PNGS + AE



- PNGS + AE*: alternating AE and PNGS
- AE identifies irrelevant actions
- PNG searches for shortcuts

ITSA* vs PNGS for Planning Tasks

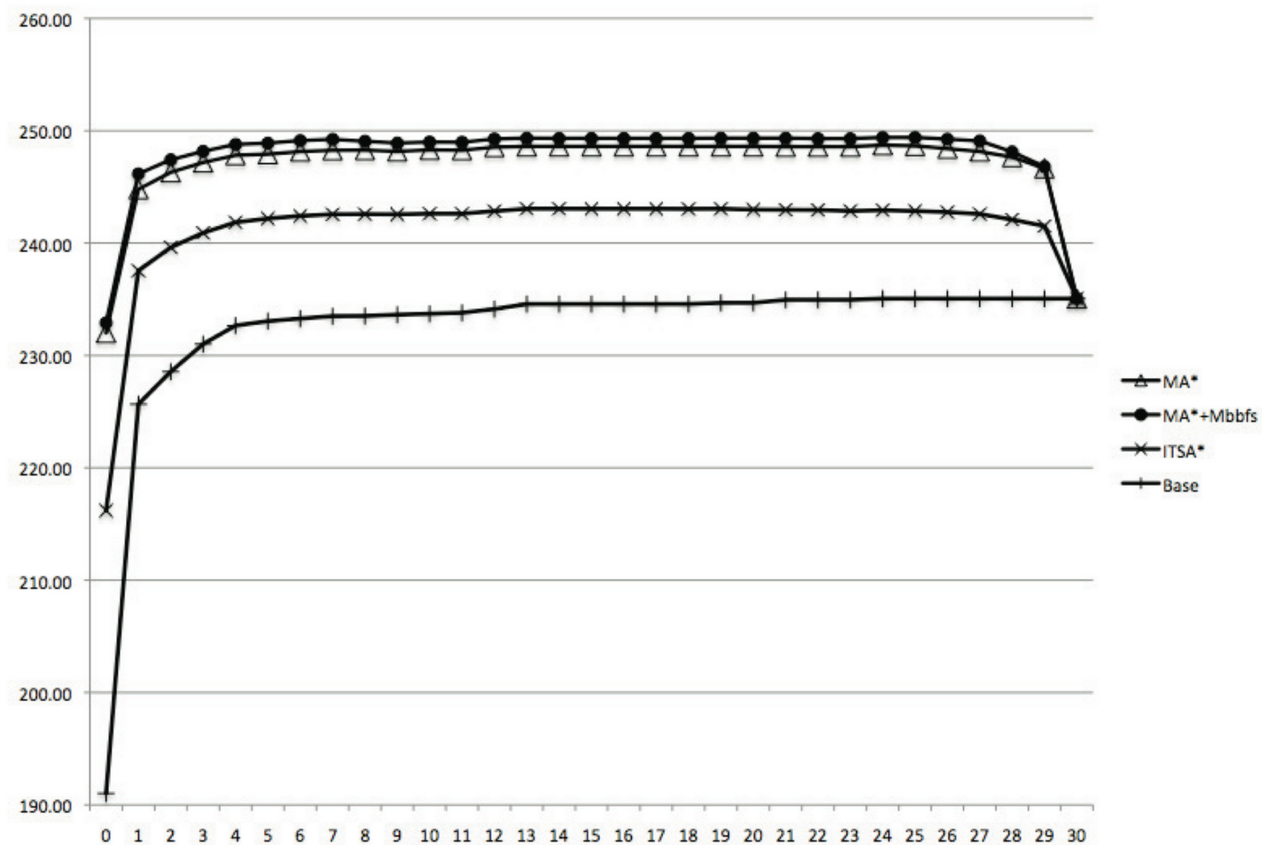


Figure 2: Total IPC-2008 score for varying cutoff times combining LAMA with M_{A*}, $M_{A*} + M_{bbfs}$ and ITSA*

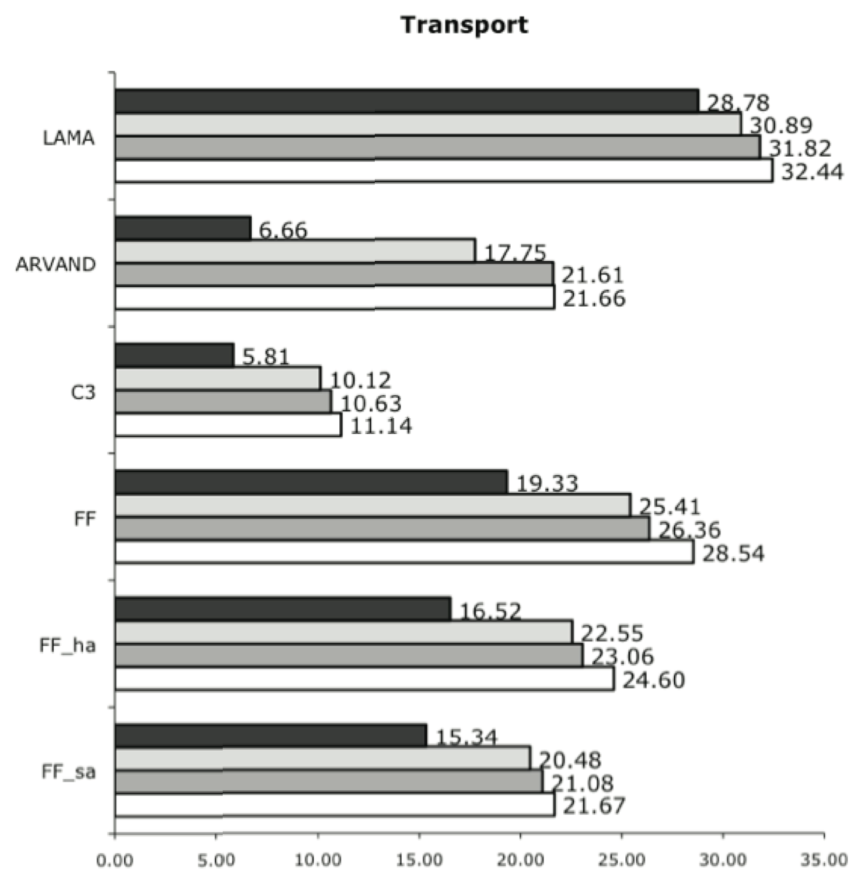
Conclusion

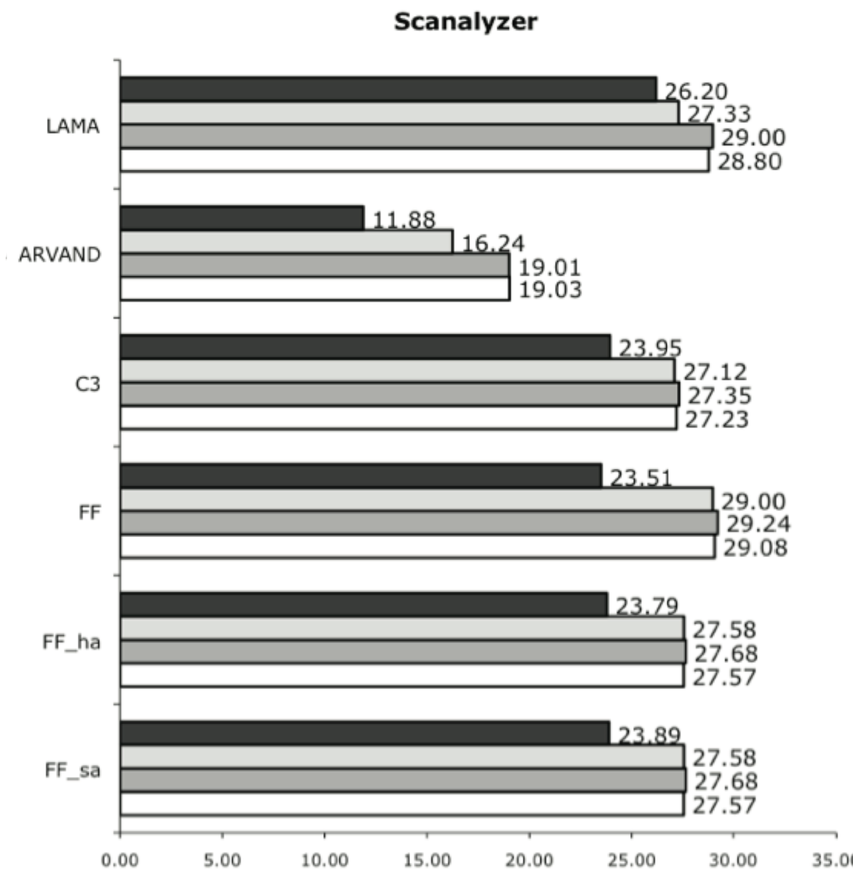
- PNGS (+AE) superior to ITSA*
- ITSA* no parameter, easy to handle
- PNGS is more flexible:
    - Appropriate search algorithms can be chosen
- PNGS always with AE:
    - AE cheap and very fast (several minutes vs 1s)

Conclusion

- Solution almost always improved
- Improvement depends on the domain
- Again: A good solution is difficult to improve
- Anytime algorithms are convenient

Domains for the Software Projects





Domains for the Software Projects