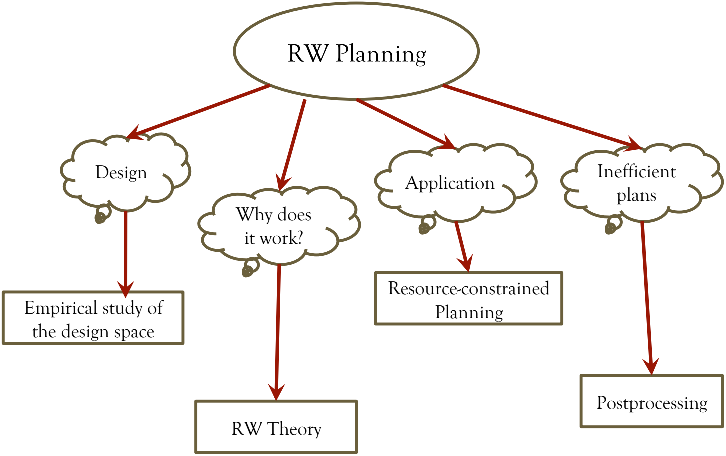
Random Walk Planning: Theory, Practice, and Application

Hootan Nakhost

University of Alberta, Canada
Google Canada since May 2013

May 9, 2012

Outline


RW Planning
Design
Why does it work?
Application
Inefficient plans
Empirical study of the design space
RW Theory
Resource-constrained Planning
Postprocessing

Automated Planning

Given a model of the world, generate a plan to achieve predefined goals

Applications

- Autonomous agents
- General solvers

Classical Representations (STRIPS)

State

Each state is a set of propositions

{On(B, A), Ontable(A), Clear(B)}

Action

Each action has preconditions, positive and negative effects

{OnTable(A), Holding(B)}

Plan

A sequence of actions that starts from the initial state and ends in $s \supseteq G$

Planning Methods

Heuristic Search

Common standard systematic search algorithms such as Greedy Best First Search (GBFS) and WA*

Contribution

A new search paradigm for satisficing planning: random walk (RW) search

Why Random Walks?

Random Walk

A sequence of randomly selected actions

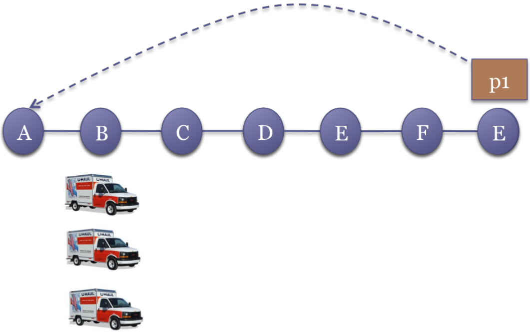
High level and Intuitive Explanations

- Escaping faster from plateaus
- More exploration
- Not wasting time in dead-ends

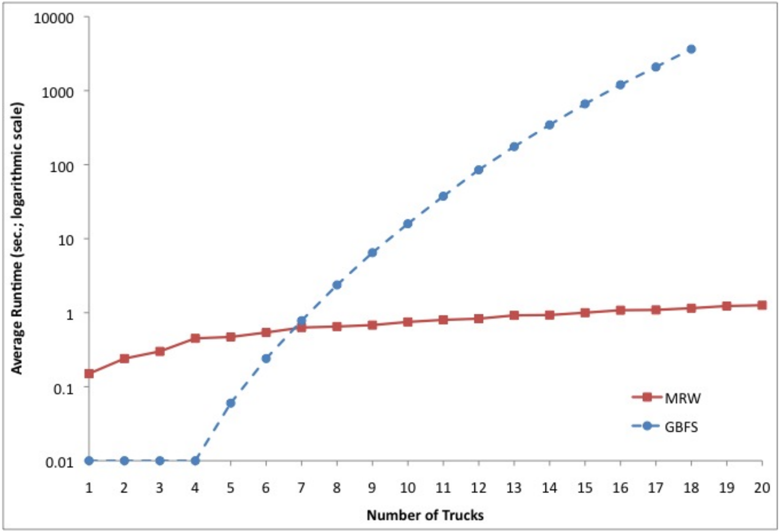
A theoretical model can explain ...

- What are the key features affecting the performance
- How we can improve the algorithms

A Motivating Example: Transportation Domain



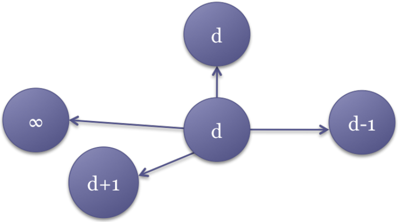
Random Walks vs. Systematic Search



Theoretical Analysis of RW Planning

Graph properties affecting RW performance

- Progress Chance(PC)
- Regress Chance(RC)
- Regress Factor(RF)



$$PC = \frac{1}{4}, RC = \frac{1}{2}, RF = \frac{RC}{PC} = 2$$

Definitions: Fairness and Hitting Time

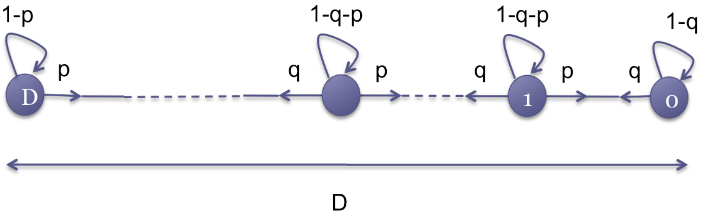
Fairness

A single state transition in the graph cannot change the goal distance by more than one unit.
Every undirected graph is a fair graph.

Hitting Time

The expected number of steps in a random walk starting from the initial state and ending in the goal for the first time.

Fair Strongly Homogenous Graph (FSHG)



p = progress chance
q = regress chance
D = largest goal distance

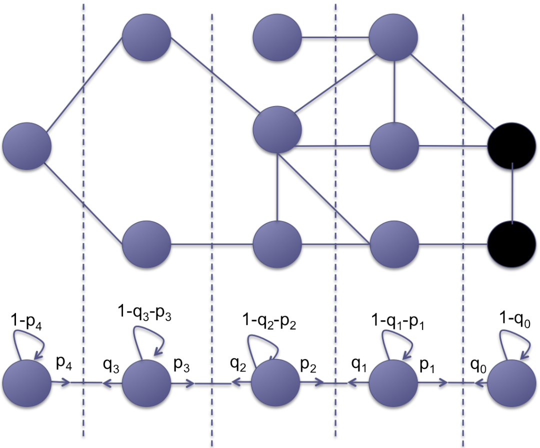
Theorem: Hitting time in FSHG

$$h_x = \begin{cases} \Theta\left(\beta_0 \lambda^D + \beta_1 d_x\right) & \text{if } q \neq p \\ \Theta\left(\alpha_1 D d_x\right) & \text{if } q = p \end{cases}$$

where

$$\lambda = \frac{q}{p}, \beta_0 = \frac{q}{(p-q)^2}, \beta_1 = \frac{1}{p-q}, \alpha_0 = \frac{1}{2p}, \alpha_1 = \frac{1}{p}$$

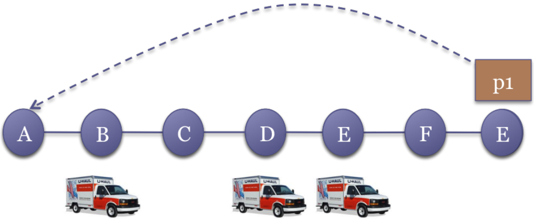
Bounds for more general graphs



q_i = maximum regress chance at the goal distance i

p_i = minimum progress chance at the goal distance i

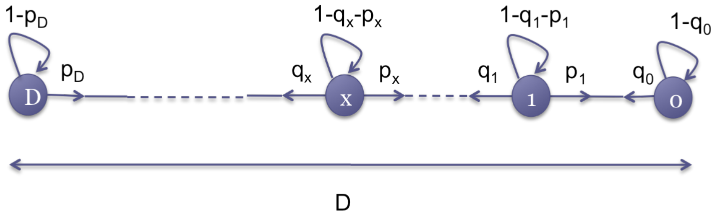
Analysis of the Transport Example



$$RC_{max} = PC_{min} = \frac{1}{2 \times |\text{trucks}|}$$

$$h_x = \frac{Dd_x}{p}$$

Fair Homogenous Graph (FHG)



p_i = progress chance at goal distance i
q_i = regress chance at goal distance i
D = largest goal distance

Hitting time in FHG

$$h_x = \sum_{d=1}^{d_x} \left(\beta_D \prod_{i=d}^{D-1} \lambda_i + \sum_{j=d}^{D-1} \left(\beta_j \prod_{i=d}^{j-1} \lambda_i \right) \right)$$

where for all $1 \leq d \leq D$,

$$\lambda_d = \frac{q_d}{p_d}, \beta_d = \frac{1}{p_d}$$

Theory for Random Walks with Restart

Restarting Random Walks

At each step with probability r restart from the initial state

Hitting Time

$$h_x \in O\left(\beta\lambda^{d_x - 1}\right)$$

where

$$\lambda = \left(\frac{q}{p} + \frac{r}{p(1-r)} + 1\right), \beta = \frac{q+r}{pr}$$

Findings

- Determined the key features of the search space affecting RW
  - Regress factor RF
  - Largest goal distance D
  - Initial goal distance d
- Provides valuable insights to design RW planners
  - Biasing action selection
  - Restarting frequency r

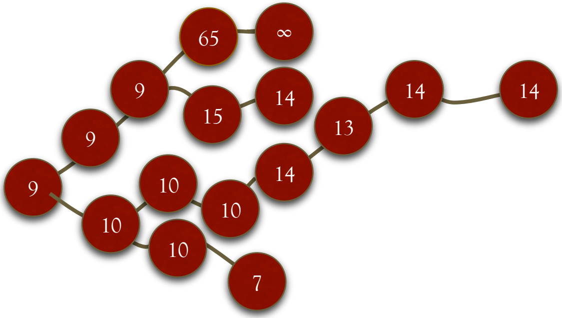
RW Search

The General Framework

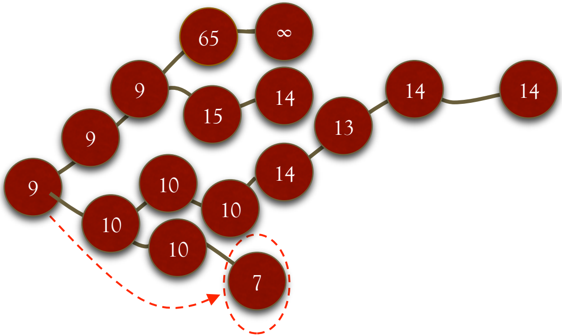
- Use forward chaining Local Search
- In each step, run random walks to find the next state
- Use restarts to recover from unpromising search regions

RWS Framework: an Illustration

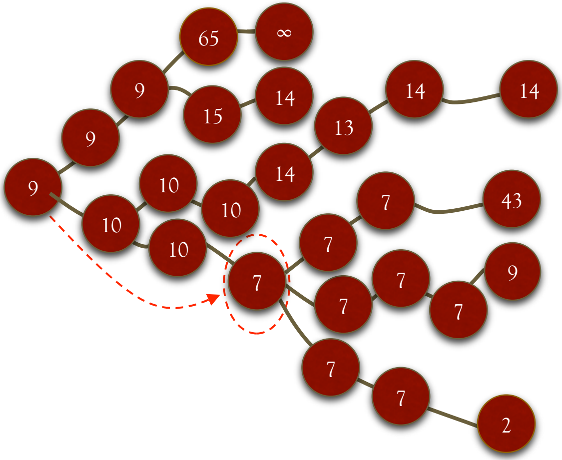
RWS Framework: an Illustration



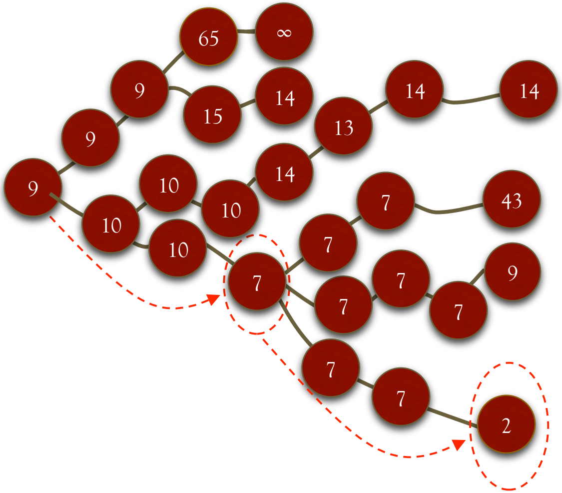
RWS Framework: an Illustration



RWS Framework: an Illustration



RWS Framework: an Illustration



A Basic RW planner

Walk Length

Use a local restarting rate r_l: at each step terminate the walk with probability r_l

Restarting

Use a restarting threshold t_g: restart the search when the last t_g walks have not reached lower heuristic

Experimental Study of the Design Space

Local Exploration

- Length of Walks
- Evaluation Rate
- Action Selection Bias

Global Exploration

- Jumping Strategies
- Restarting Strategies

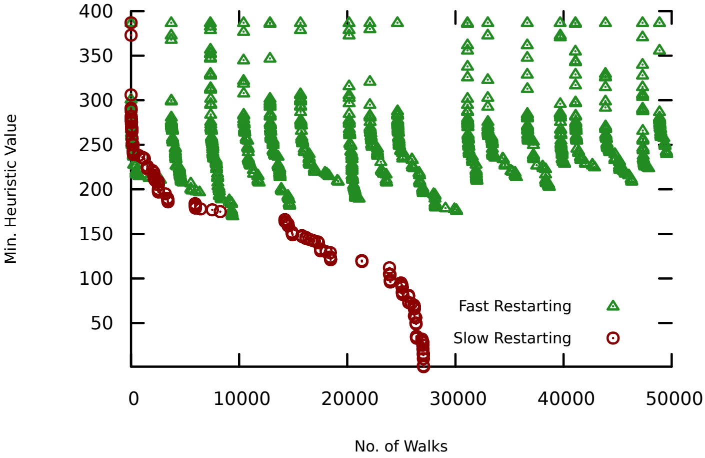
Heuristic function

- Type of the heuristic function
- The accuracy of the heuristic function

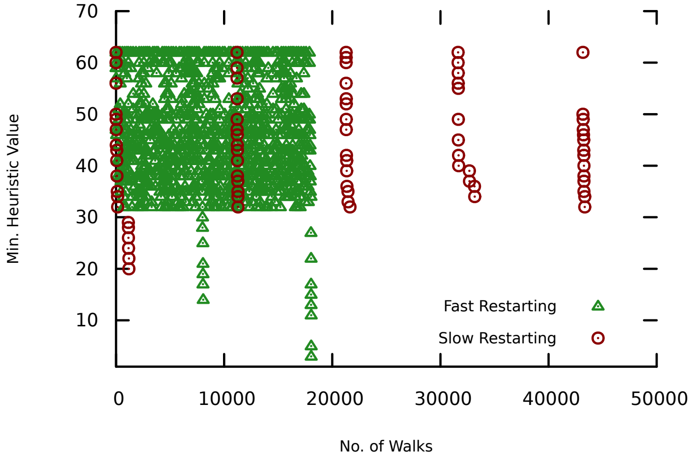
Two Practical Outcomes

- Learning systems that adapt parameters to the input problem
- Effective Biasing techniques

The Effect of Restarting Threshold: Elevators 03

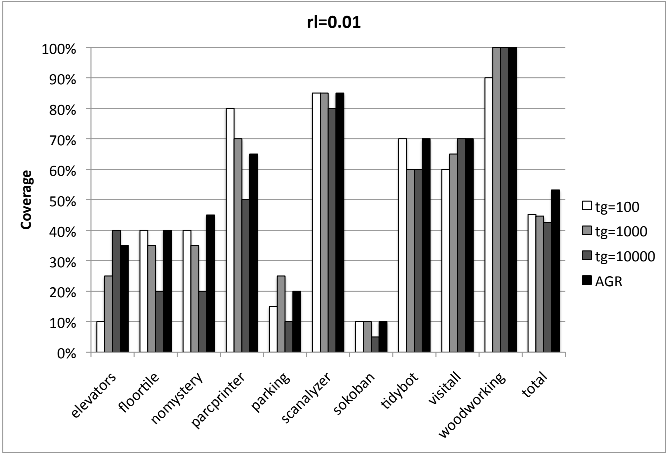


The Effect of Restarting Threshold: Floortile 01

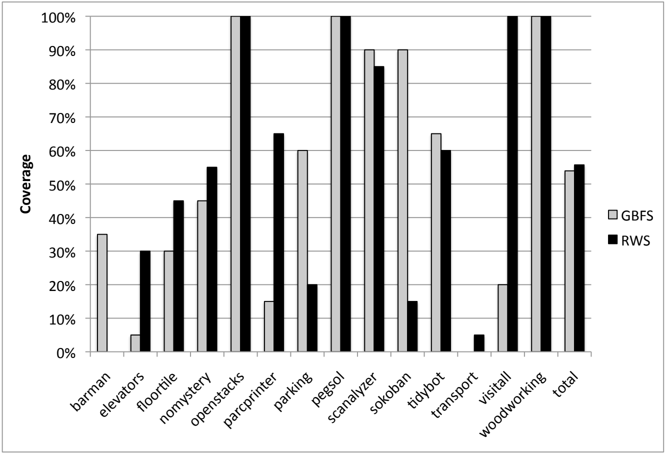


Adaptive Global Restarting (AGR)

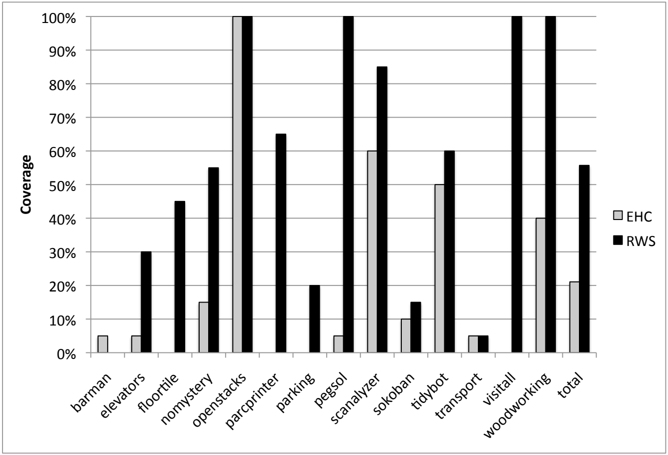
- Let V_w be the average heuristic improvement per walk
- AGR continually estimates V_w and sets $t_g = \frac{h_0}{V_w}$



Comparison with GBFS



Comparison with EHC



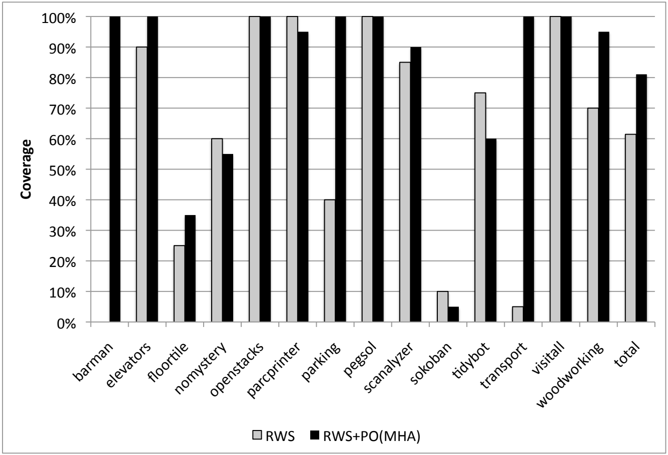
Biasing Action Selections

Monte Carlo Helpful Actions (MHA)

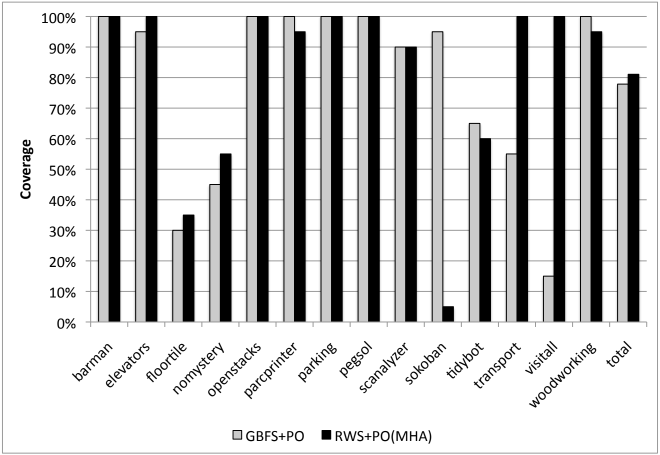
MHA gives a higher priority to *preferred operators*.

$$P(a, s) = \frac{e^{Q(a)/T}}{\sum_{b \in A(s)}^{n} e^{Q(b)/T}}$$

MHA vs. Uniform Action Selection



MHA vs. GBFS+PO



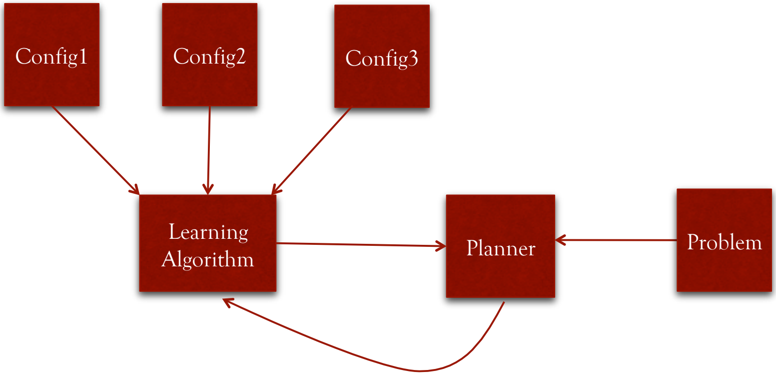
Building a Planning System

- Combine several techniques that complement each other

Examples

- Multiple heuristics in LAMA and Fast Downward
- Multiple search strategies in Fast Forward and FD Stone Soup

Learning the Best Configuration



Comparing Arvand-2013 with Top Satisficing Planners

Table: IPC problems without Derived Predicates

No. of Problems	Arvand-2013	LAMA-2011	FDFSS2	Probe	Roamer
1661	**1552**	1540	1533	1422	1507

Table: All IPC problems

No. of Problems	Arvand-2013	LAMA-2011	FDFSS2	Probe	Roamer
1857	1666	1659	**1668**	–	1635

The Gap Between RW and Systematic Planning

Domains	Arvand-2013	LAMA-2011
Airport (50)	**44**	31
Notankage (50)	**50**	44
Sokoban (20)	1	**19**
Storage (30)	**30**	19
Tankage (50)	**44**	41
Woodworking (30)	14	**20**
Philosophers (48)	**44**	34
PSR Large (50)	19	**31**
PSR Middle (50)	43	**50**

Reasoning about Resources

Examples of limited resources

Fuel, energy, money, time

Model: not replenishable resources

- Initial supply
- Some actions consume resources

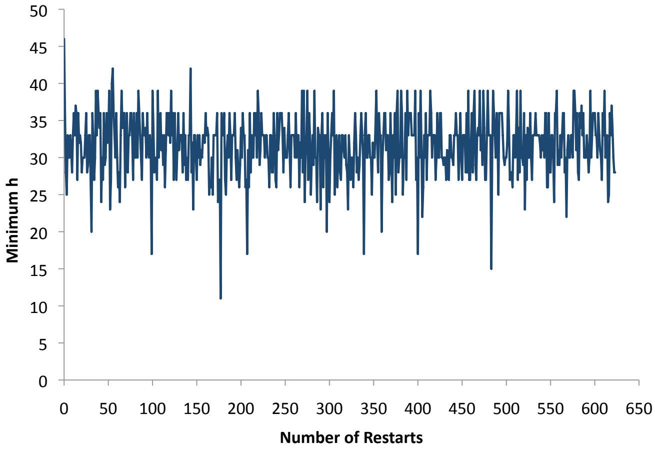
Limitation of the Current Methods

- Relaxation heuristics do not model resource consumption at all
- Greedy search algorithms add more problems

Improvements to Arvand for RCP

- Smart Restarting (SR)
- On-path Search Continuation (OPSC)

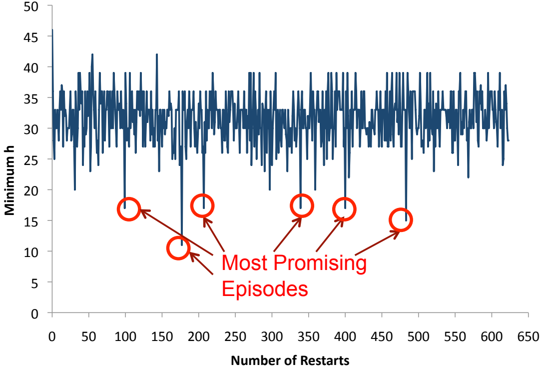
Basic Restarting in an Example



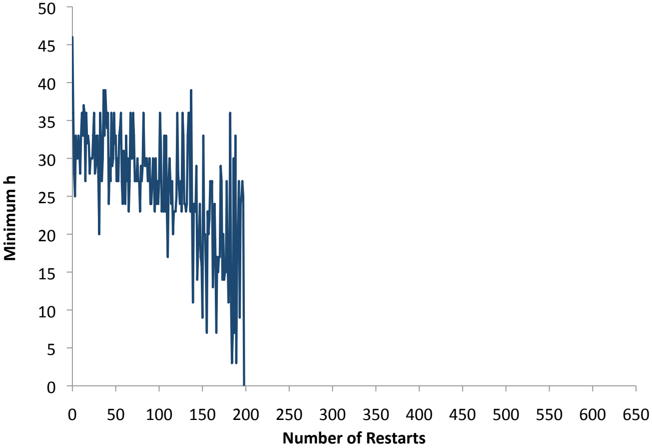
Smart Restarting

Algorithm

- Maintain a pool of most promising episodes performed
- When an episode gets stuck restart from a state visited in an episode in the pool



Smart Restarting in an Example



How to test RCP planners?

Performance as a function of constrainedness

Resource constrainedness C (Hoffmann et. al. IJCAI-2007)

$$C = \frac{\textit{initial supply}}{\textit{minimum need}}$$

The closer C is to 1, the more constrained is the problem.

My Contributions

- Extended the definition of C to multiple resources
- Developed two new benchmarks for RCP

Experiments

3 RCP Domains

NoMystery, Rovers, TPP

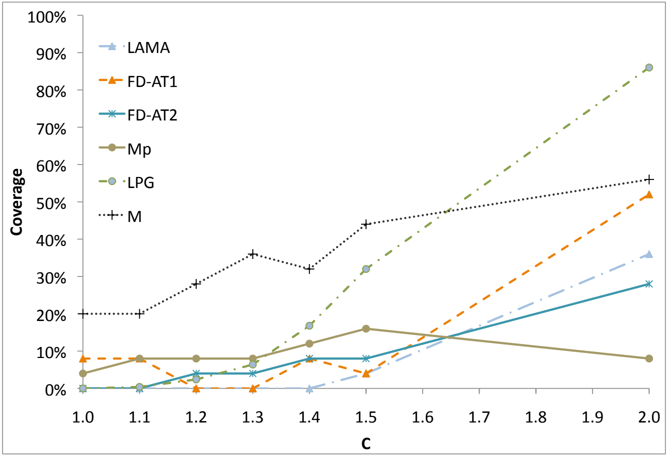
8 Satisficing Planners

Arvand, FD-AT1, FD-AT2, LAMA, FF, LPG, M, Mp, LPRPGP

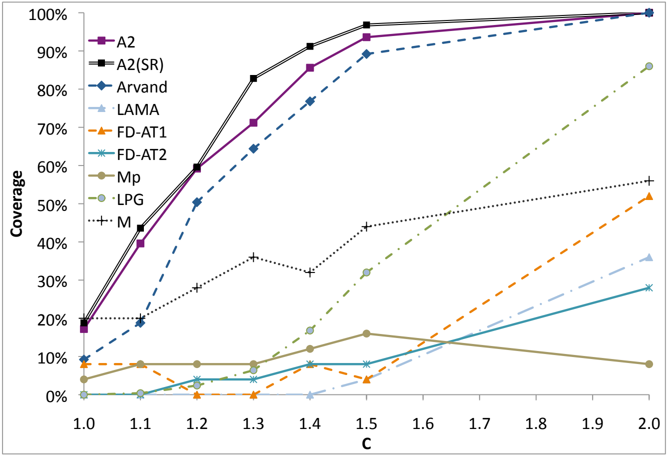
5 Optimal Planners

Num-2-sat, LM-cut, Merge and Shrink, Selmax, FD-AT-OPT

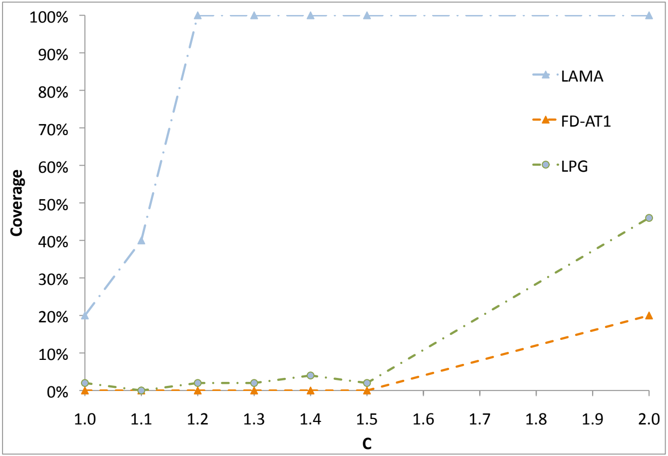
Results: Rovers, small



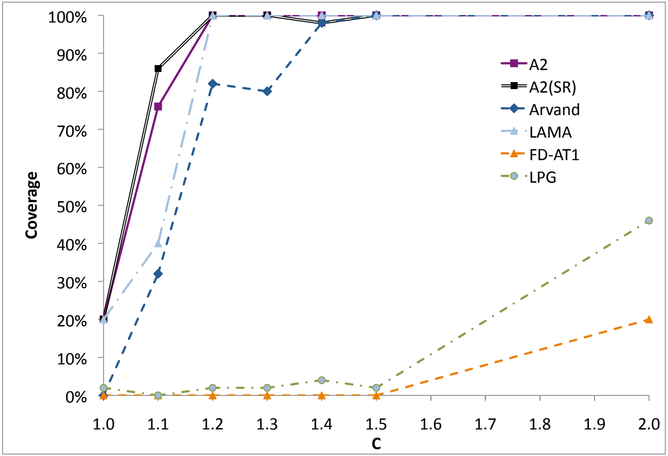
Results: Rovers, small



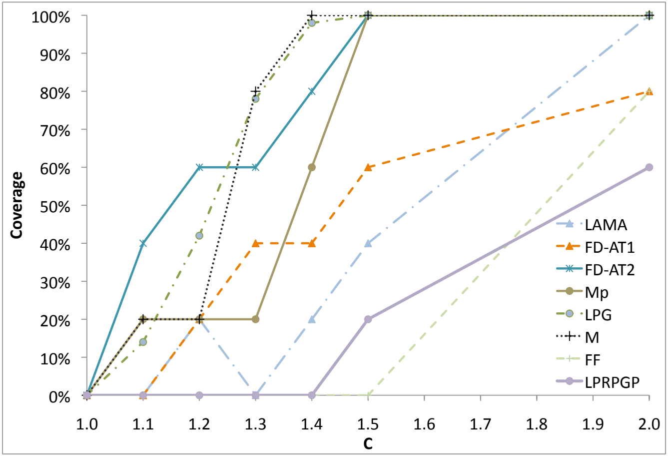
Results: Rovers, large



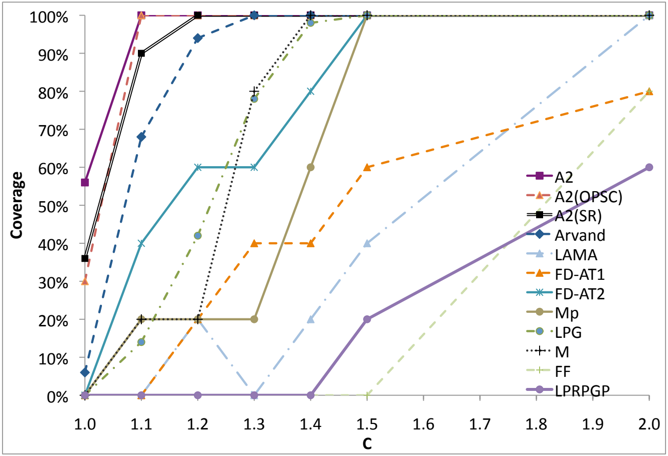
Results: Rovers, large



Results: NoMystery, large



Results: NoMystery, large



Plan Improvement

RW planning can generate bad-quality solutions

Idea

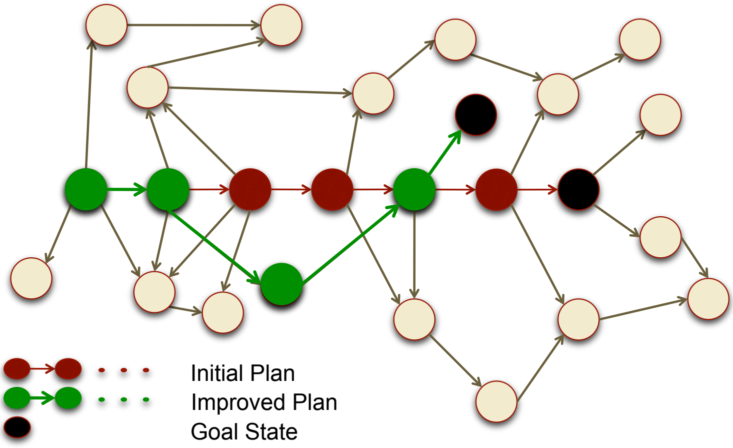
Develop fast post-processing techniques to improve the solutions

Outcome: Aras

A postprocessor that works well for a wide range of planners

- Even for those like LAMA that are well-designed to generate good-quality solutions

Plan Neighborhood Graph Search (PNGS)



Anytime PNGS

- Iteratively increase the expansion limit
- Each iteration starts with last plan generated in previous iterations

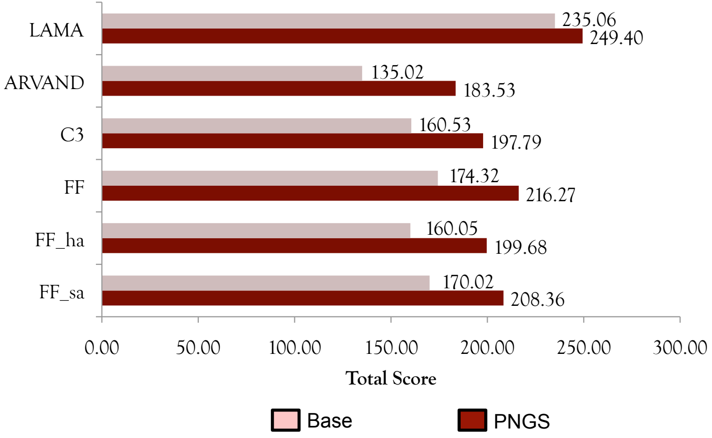
Experiments

- Compare state-of-the-art planners with and without plan improvement on IPC domains
- Scoring function: the cost of the best plan produced by any planner divided by the cost of the generated plan
- Issue: how to divide time between planner and postprocessor

Cutoff Time

- Run the planner until a cutoff time is reached
  - If no solution is found, keep running until the first solution is found
- Use the postprocessor to improve the best generated plan

IPC-2008 PNGS



Integration of Arvand-2013 and Aras

- Repeat until the time limit (30 min.) is reached:
  - Run Arvand-2013 until a solution *s* is found
  - Run Aras to improve *s* until a memory/time limit (2 GB) is reached
- The cost of the best previous plan is used for prunning
- Report the best plan found as the result

Arvand-2013 vs. Top Planner (Solution Quality)

Domain	Arvand-2013	LAMA-2011	FDFSS2	FDFSS1	Roamer
Scanalyzer	16.17	15.63	16.91	**17.70**	15.46
Pegsol	**19.88**	**19.88**	16.02	14.70	18.11
Floortile	5.00	4.46	**6.35**	5.44	1.63
Tidybot	11.22	14.53	11.23	**14.82**	13.03
Nomystery	**13.39**	11.33	10.80	13.33	9.51
Transport	12.10	12.39	9.14	9.46	**14.39**
Parcprinter	**19.00**	18.87	18.95	16.65	5.83
Elevators	8.64	10.62	8.70	**12.41**	11.74
Visitall	11.89	15.84	3.08	2.77	**16.89**
Parking	10.11	**16.96**	12.40	8.72	8.34
Woodworking	12.75	14.23	18.42	**18.56**	11.78
Barman	**19.93**	17.15	10.86	14.31	15.30
Sokoban	1.00	**16.28**	13.90	15.88	13.22
Openstacks	11.83	**18.36**	11.11	12.68	17.57
Total	172.88	**206.52**	167.88	177.43	172.80

Random Walk Planners

- Arvand-2009: Establishing the foundation
- Arvand-RC: Using RW Search for RCP
- Arvand-2011: Learning the Best Configuration and Using Aras
- Arvand-LS: RandomWalks with Memory
- ArvandHerd: Parallel portfolio

Contributions

RW search as an effective framework for satisficing planning

- A theoretical framework for studying RW search
  - Determined key features affecting RW
  - Explained where and why RW exploration is effective
- A detailed experimental study of design space
  - Built effective learning systems that adapt parameters
  - Built efficient biasing techniques
  - Gained valuable insights regarding the effects of different parameters

Contributions

- Application of RW search to RCP
  - Extended the definition of C to multiple resources
  - Developed of new benchmarks
  - Significantly improved the state of the art
- Aras: a very effective postprocessing system
- Several strong planning systems
  - Arvand-2009: Establishing the foundation
  - Arvand-2011: Configuration learner and Aras
  - Arvand-2013: Empirical study of the design space
  - Arvand-RC: Using RW search for RCP
  - Arvand-LS: RW with memory
  - ArvandHerd: Parallel portfolio

Thank you for your attention!