Predicting Optimal Solution Cost with Bidirectional Stratified Sampling

Levi Lelis
Computing Science Dept.
University of Alberta

Joint work with:

Roni Stern and Ariel Felner
Information Systems Engineering
Ben-Gurion University

Sandra Zilles
Computer Science Dept.
University of Regina

Robert Holte
Computing Science Dept.
University of Alberta

The Problem

- Given:
  - a start state and a goal state of a state space problem.
- Efficiently predict the optimal solution cost from start to goal.

Motivation

- A path from start to goal is not always required. Sometimes the solution cost suffices.
  - Bidding problem.

- Accurate predictions of the solution cost can be used to enhance search.
  - Find the w-value for WIDA* (SoCS 2011).
  - Set the upper bound for Potential Search (SoCS 2011).
  - Learn strong heuristic functions (SoCS 2012).

Cost Prediction vs. Heuristics

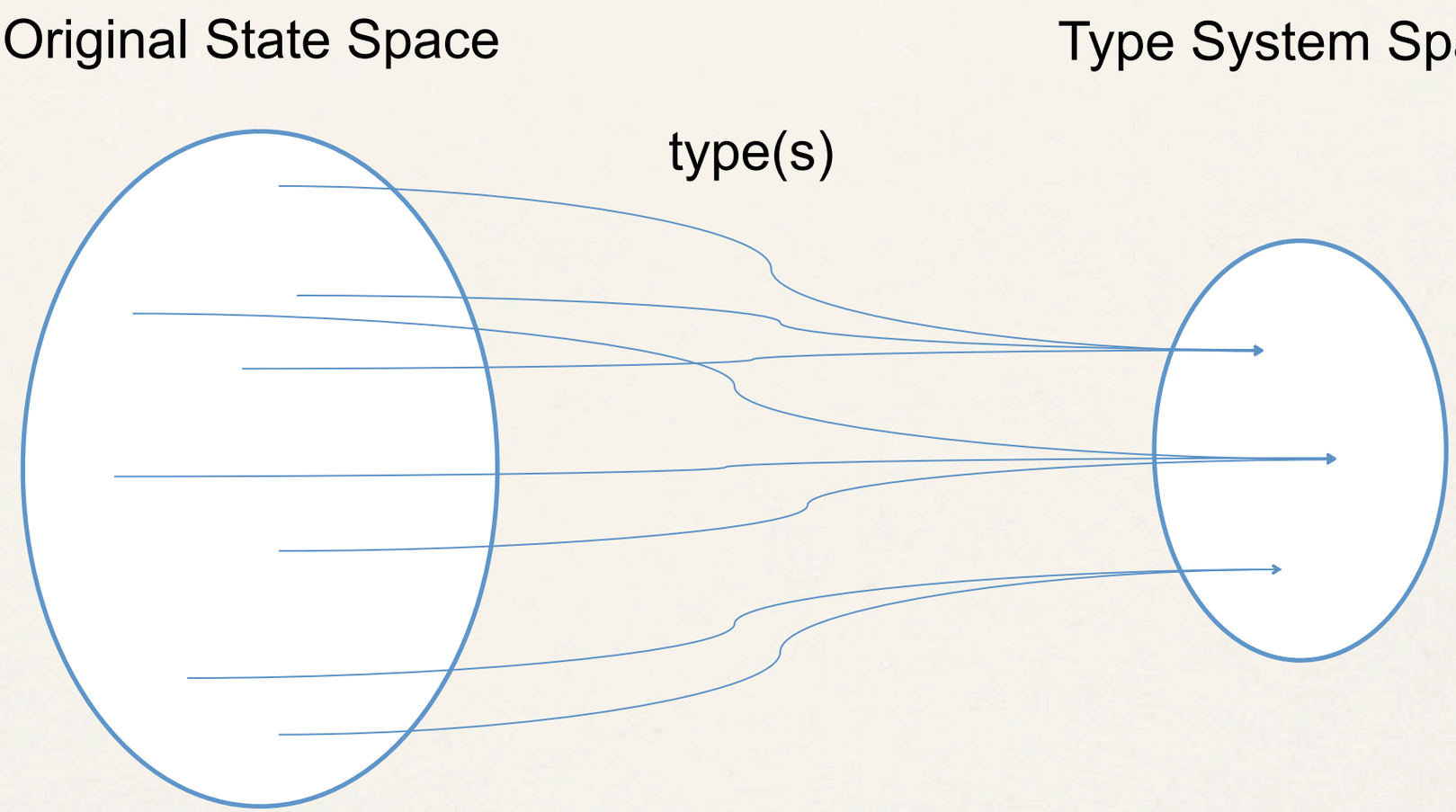
- Heuristic functions also estimate the solution cost. However,
  - Often are biased to never overestimate the optimal solution cost.
    - Poor Estimates.
- Need to be fast to guide search.
  - We expect more accurate estimates by allowing predictors more computation time.
- Our measure of effectiveness is accuracy, not search speed.

Existing Algorithm

- Solution Cost Predictor (SCP) -- with Roni Stern and Shahab Jabbari Arfaee (SoCS'11).
  - SCP has problems scaling to large state spaces.

Definition

Type System for States



Type System for States

- A Type System might consider:
  - the heuristic value.
  - other information about the states.

Illustrative Example - 8-puzzle

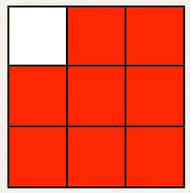
- Heuristic = distance of the blank from its goal position (top left).

- The other information indicates if the blank is in a corner, egde of middle position

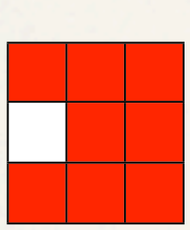
	1	2
3	4	5
6	7	8

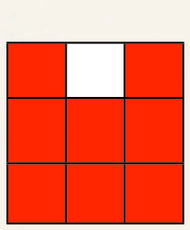
C	E	C
E	M	E
C	E	C

Types

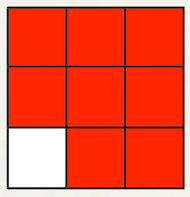


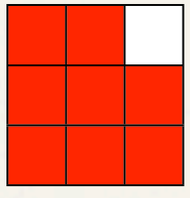
(0,C)



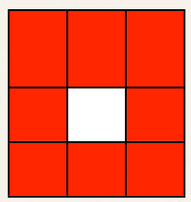


(1,E)

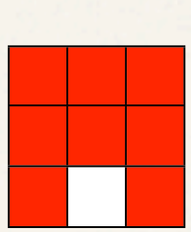


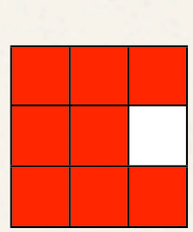


(2,C)

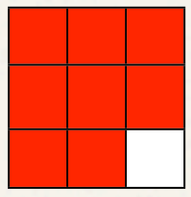


(2,M)





(3,E)



(4,C)

The Algorithm

Stratified Sampling

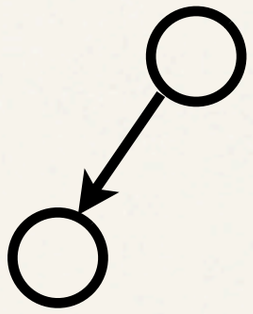
- Chen's Stratified Sampling (SS) was designed for estimating the size of backtrack search trees.

Stratified Sampling

- Chen's Stratified Sampling (SS) was designed for estimating the size of backtrack search trees.

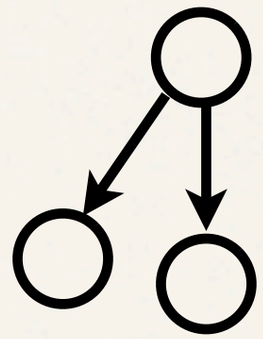
Stratified Sampling

- Chen's Stratified Sampling (SS) was designed for estimating the size of backtrack search trees.



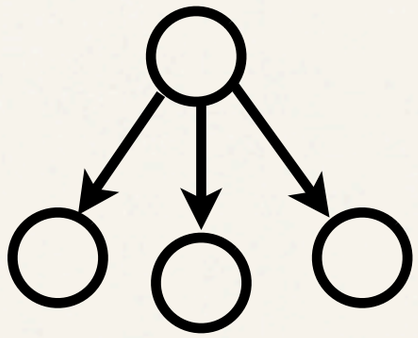
Stratified Sampling

- Chen's Stratified Sampling (SS) was designed for estimating the size of backtrack search trees.



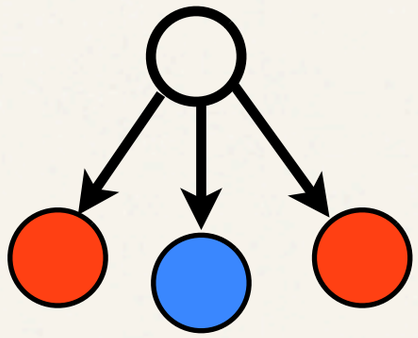
Stratified Sampling

- Chen's Stratified Sampling (SS) was designed for estimating the size of backtrack search trees.



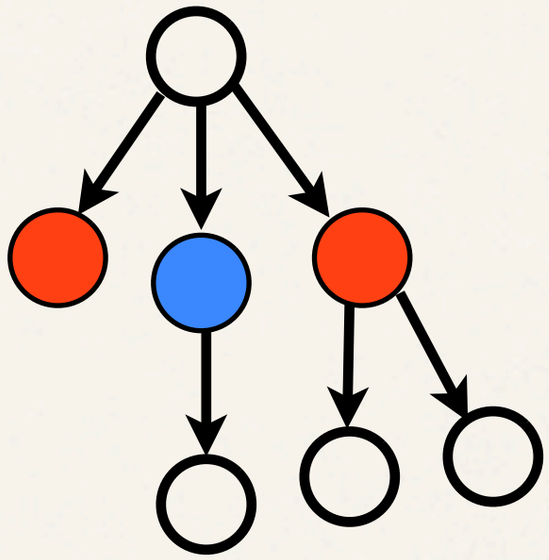
Stratified Sampling

- Chen's Stratified Sampling (SS) was designed for estimating the size of backtrack search trees.



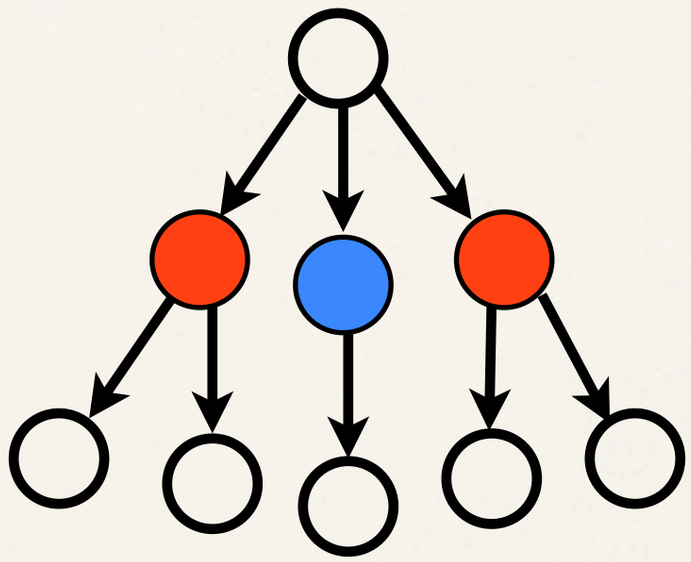
Stratified Sampling

- Chen's Stratified Sampling (SS) was designed for estimating the size of backtrack search trees.



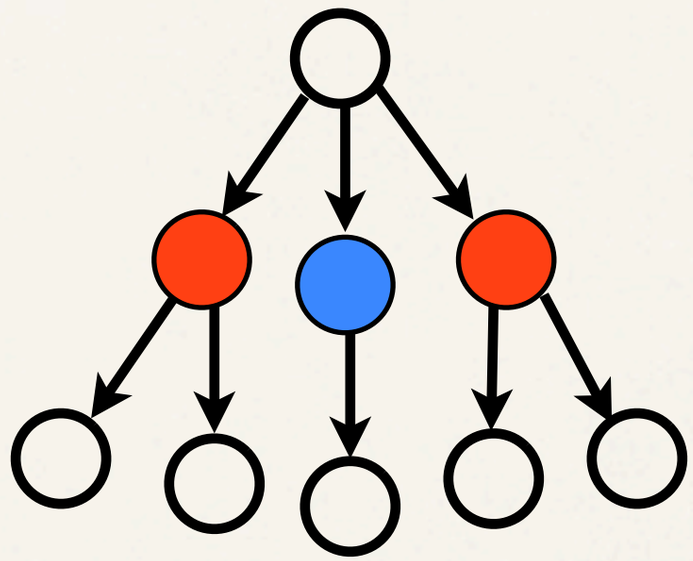
Stratified Sampling

- Chen's Stratified Sampling (SS) was designed for estimating the size of backtrack search trees.



Stratified Sampling

- Chen's Stratified Sampling (SS) was designed for estimating the size of backtrack search trees.



- SS efficiently estimates the size of backtrack search trees.

SS for Predicting Optimal Cost?

- Chen was aiming at predicting the size of backtrack search trees.
- In theory SS can be used to measure any property in the search tree.
- Is SS able to accurately predict the optimal solution cost?

First Attempt

- Run SS until the goal is found.
  - An actual path from start and goal will be found.
- Initial results on the 24-puzzle were not promising (solution length was more than 2x optimal cost).

Bidirectional SS (BiSS)

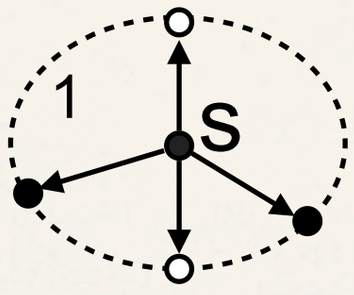
- BiSS predicts the optimal solution cost by running SS from both directions, forward from the start and backward from the goal.

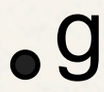
Bidirectional SS (BiSS)

- BiSS predicts the optimal solution cost by running SS from both directions, forward from the start and backward from the goal.

Bidirectional SS (BiSS)

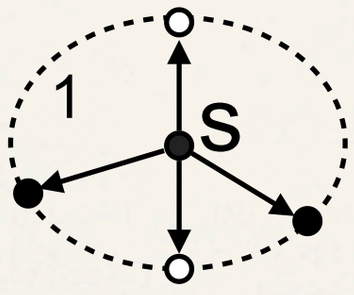
- BiSS predicts the optimal solution cost by running SS from both directions, forward from the start and backward from the goal.

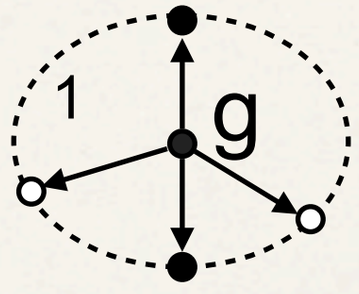




Bidirectional SS (BiSS)

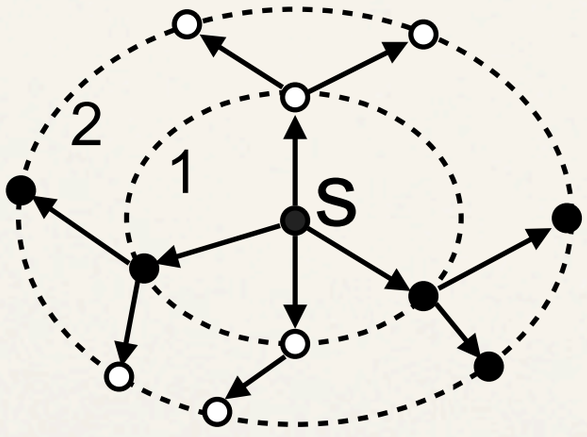
- BiSS predicts the optimal solution cost by running SS from both directions, forward from the start and backward from the goal.

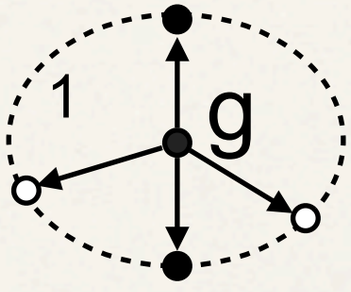




Bidirectional SS (BiSS)

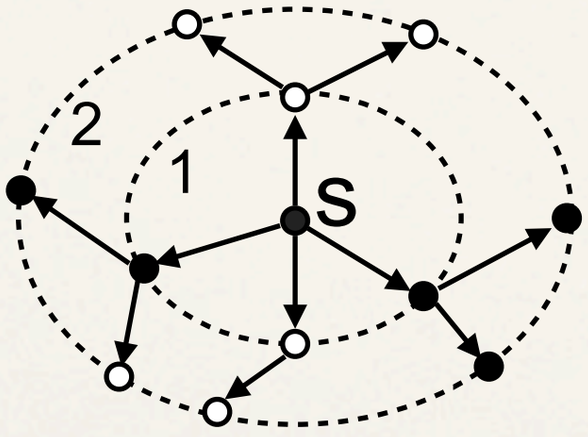
- BiSS predicts the optimal solution cost by running SS from both directions, forward from the start and backward from the goal.

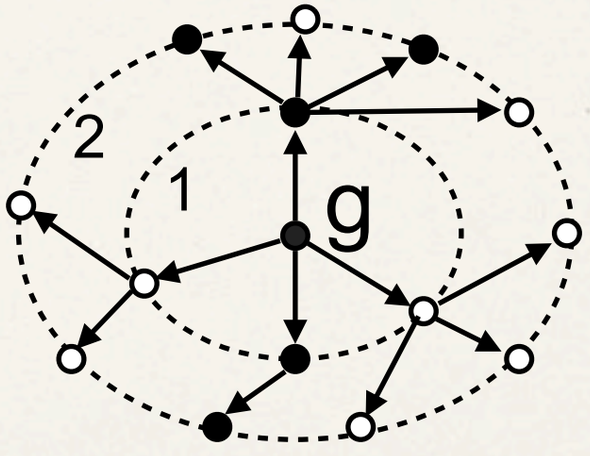




Bidirectional SS (BiSS)

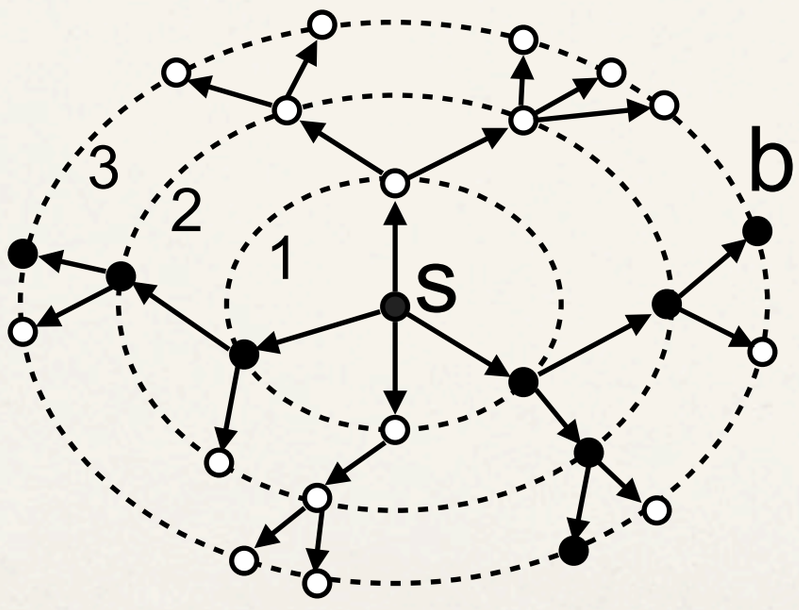
- BiSS predicts the optimal solution cost by running SS from both directions, forward from the start and backward from the goal.

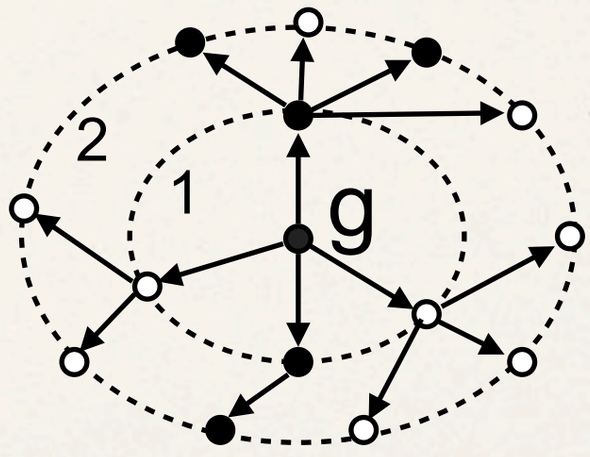




Bidirectional SS (BiSS)

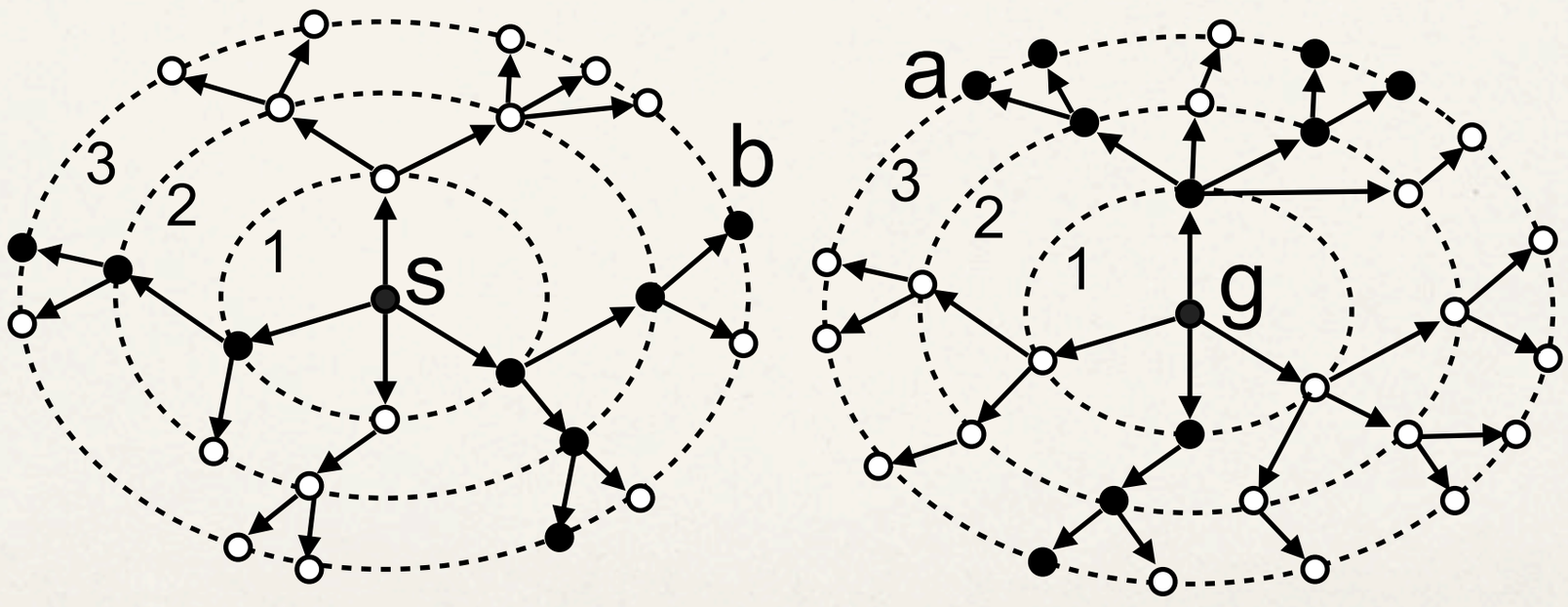
- BiSS predicts the optimal solution cost by running SS from both directions, forward from the start and backward from the goal.





Bidirectional SS (BiSS)

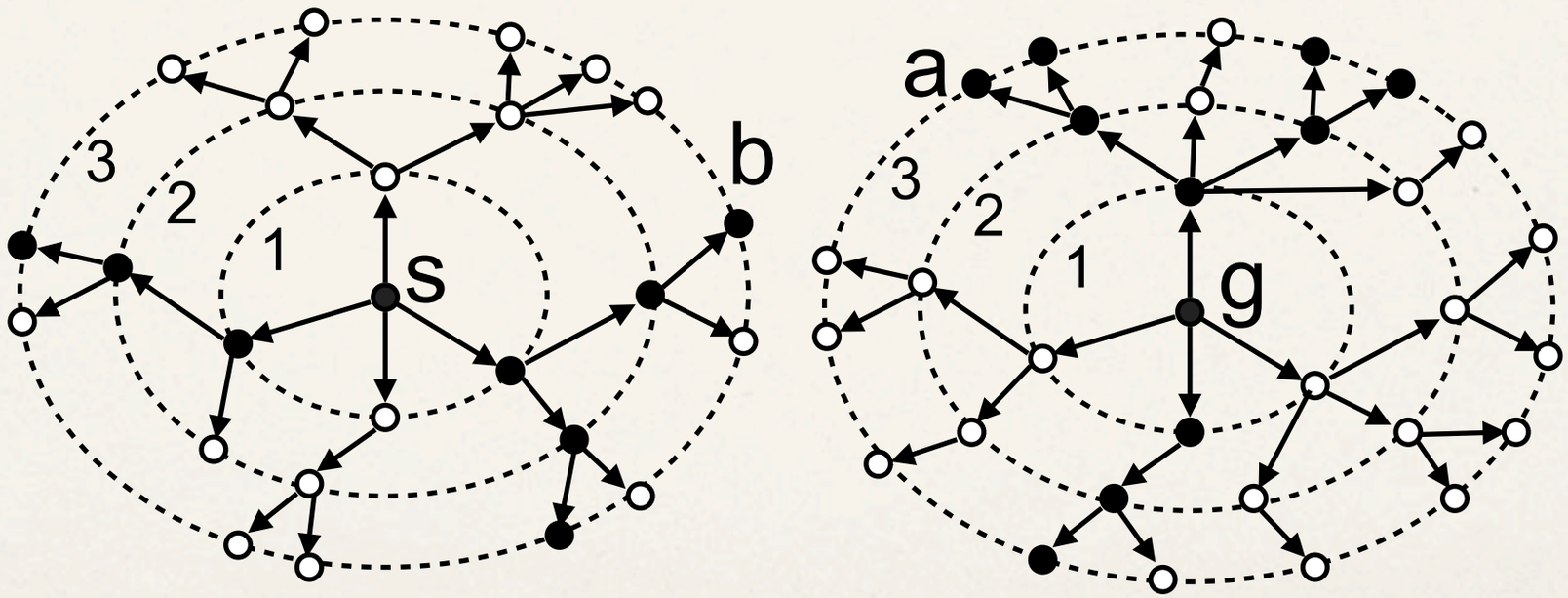
- BiSS predicts the optimal solution cost by running SS from both directions, forward from the start and backward from the goal.



Bidirectional SS (BiSS)

- BiSS predicts the optimal solution cost by running SS from both directions, forward from the start and backward from the goal.

When to Stop?



Stopping Condition

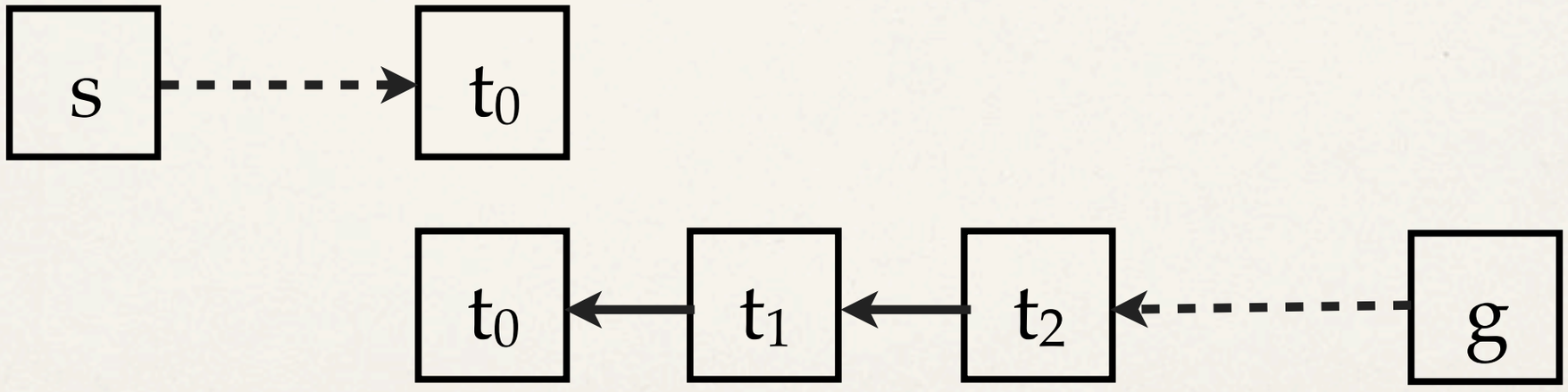
- Stopping when the intersection of types of the two frontiers is not empty will often underestimate the actual solution cost.
- A more elaborated stopping condition is required.

Stopping Condition

- Stopping when the intersection of types of the two frontiers is not empty will often underestimate the actual solution cost.
- A more elaborated stopping condition is required.

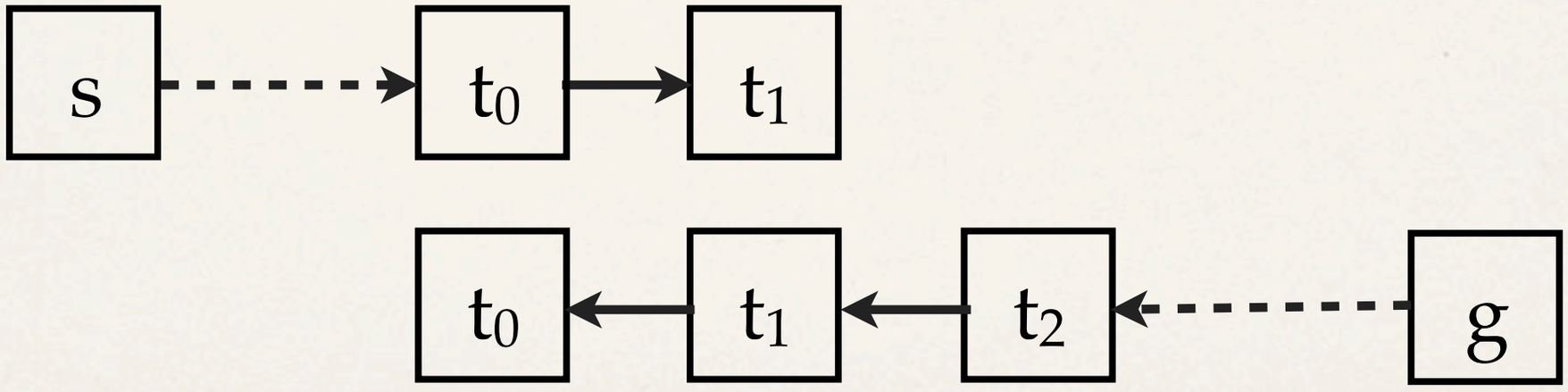
Stopping Condition

- Stopping when the intersection of types of the two frontiers is not empty will often underestimate the actual solution cost.
- A more elaborated stopping condition is required.



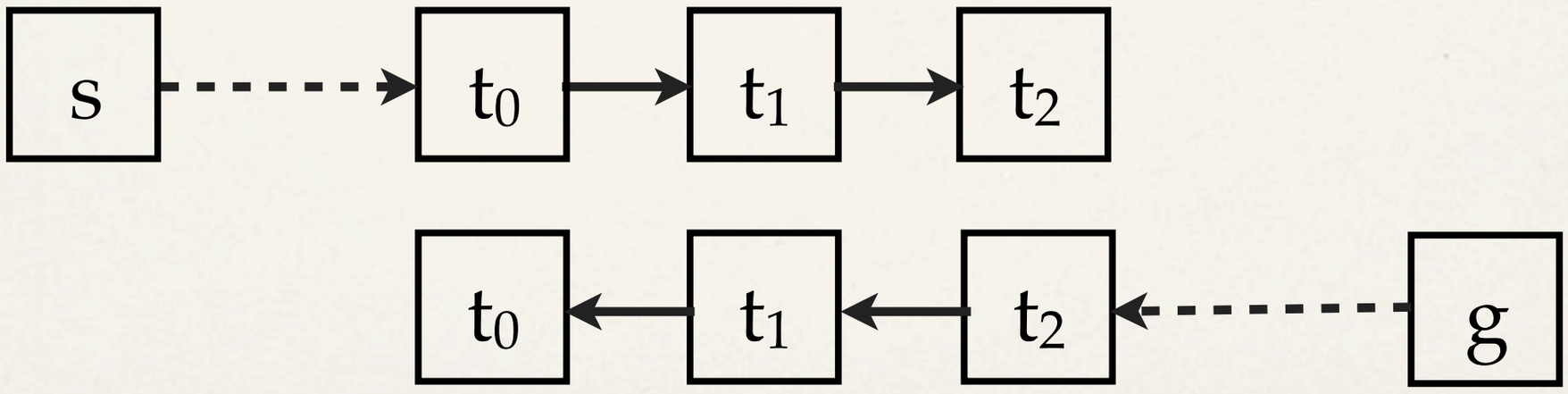
Stopping Condition

- Stopping when the intersection of types of the two frontiers is not empty will often underestimate the actual solution cost.
- A more elaborated stopping condition is required.



Stopping Condition

- Stopping when the intersection of types of the two frontiers is not empty will often underestimate the actual solution cost.
- A more elaborated stopping condition is required.



Theoretical Result

- In the limit as the number of probes goes to infinity BiSS predictions are guaranteed to be perfect.

Empirical Results

BiSS vs. SCP – 15-Puzzle

	Error		
Cost	**BiSS**	**SCP**	**Bootstrap**
51	0.07	**0.03**	0.07
52	0.07	**0.03**	0.07
53	0.07	**0.03**	0.06
54	0.07	**0.03**	0.07
55	0.06	**0.03**	0.08
56	0.07	**0.04**	0.07

BiSS vs. SCP – 15-Blocks World

	Error		
Cost	**BiSS**	**SCP**	**Bootstrap**
20	**0.01**	0.09	0.20
21	**0.02**	0.12	0.15
22	**0.02**	0.14	0.12
23	**0.01**	0.17	0.08
24	**0.01**	0.18	0.06
25	**0.01**	0.20	0.04

BiSS – 20-Blocks World

	Error	
Cost	**BiSS**	**Bootstrap**
28	**0.02**	0.18
29	**0.01**	0.15
30	**0.02**	0.13
31	**0.01**	0.10
32	**0.01**	0.08

BiSS – 24-Puzzle

	Error	
Cost	**BiSS**	**Bootstrap**
100	**0.03**	0.08
102	**0.04**	0.08
104	**0.03**	0.08
106	**0.04**	0.08
108	**0.03**	0.08

BiSS – Large Sliding-Tile Puzzles

	6x6	7x7	8x8
Time (minutes)	6	18	80
Avg. Predicted Cost	172	280	423
Polynomial Fit	171	268	397

Conclusions

- BiSS efficiently and accurately predicts the optimal solution cost (SCP **does** that too!).

- BiSS scales to very large state spaces (well, SCP **does not** scale very well).

- BiSS **might** require (SCP **does not** require these)

  - a goal state to be available, and

  - invertible operators to be available.