

Heuristic Search for the Analysis of Graph Transition Systems

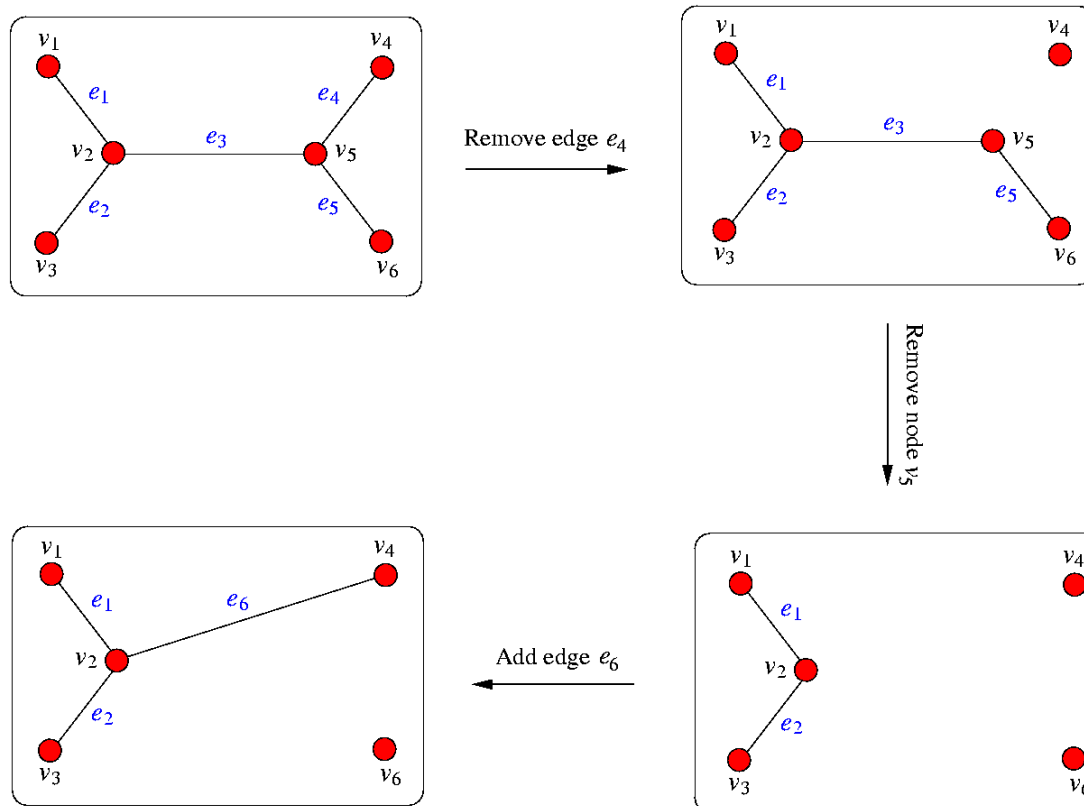


Stefan Edelkamp
Shahid Jabbar

Computer Science Department
University of Dortmund, Dortmund, Germany

Alberto Lluch-Lafuente
Empoli, Italy

Graph Transition Systems



Graph Transition Systems

- A *graph transition system* (GTS) is a pair $\langle M, g \rangle$, where
 - M is a **weighted transition system** and
 - g is a **partial graph morphism**.
- The weights of a transition system are modeled by a generalized **cost-algebra** based on monoids.
- Applications: **Biology** – Changing molecular structure, **Networks** – Clients appearing and disappearing...

Objectives

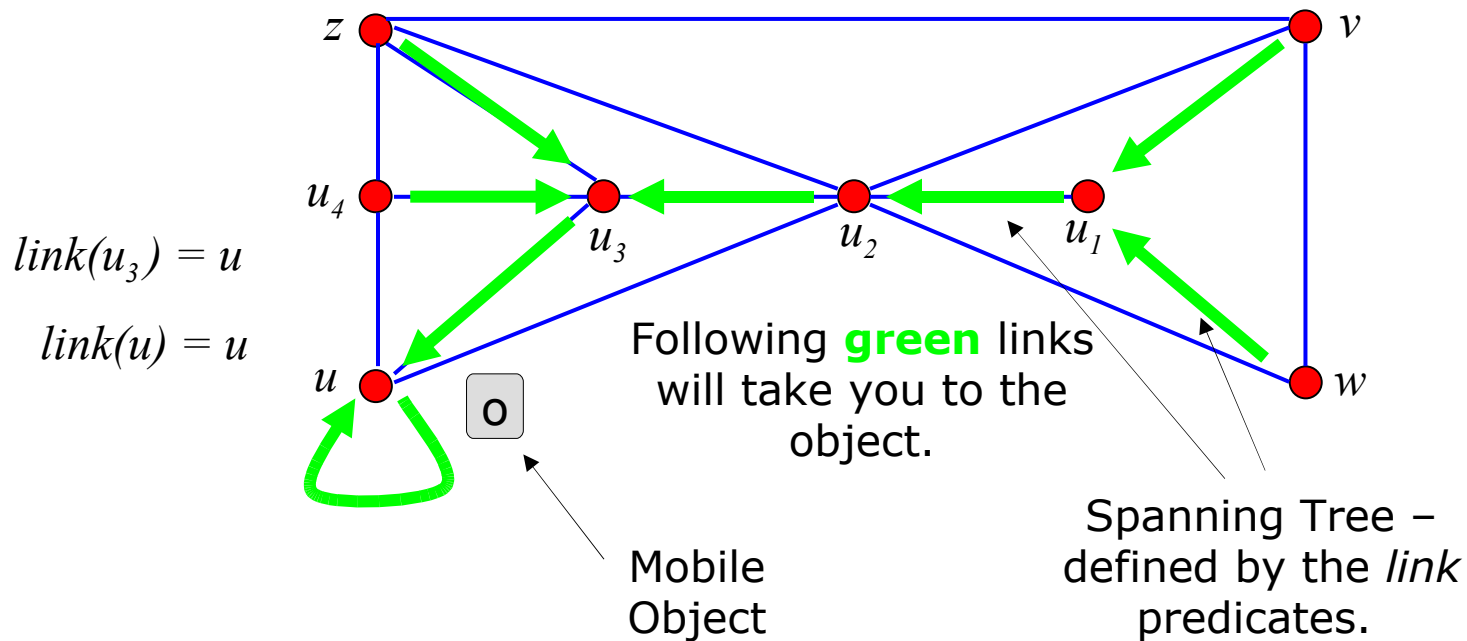
- How to model-check GTS?
 - How to deal with flexible goals?
 - How to accelerate the process?
- Solution: Use **directed model checking** heuristics.

Outline

- Arrow Distributed Directory Protocol – An example of GTS.
- Cost Algebra
- Heuristic Search
- Heuristics for Graph Transition Systems
- Experimental Results
- Conclusions

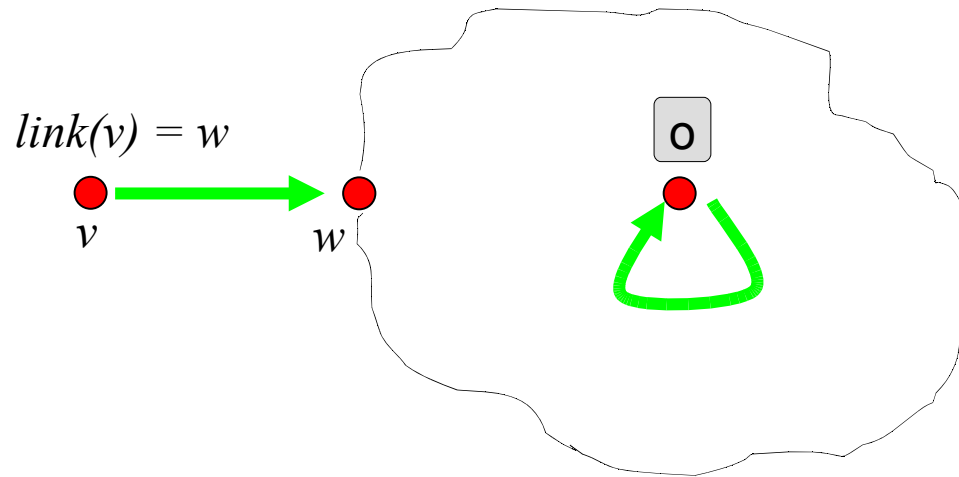
The Arrow Distributed Directory Protocol (Demmer and Herlihy)

- Based on the idea of a trail of pointers
- Distributed Network $G = (V, E, w)$



Properties of the Protocol

- **If** $link(v) = v$ (self-loop) \Rightarrow The object either resides at v , or will soon reside at v .
- **Else**, the object resides some where in the region of the directory containing $link(v)$.

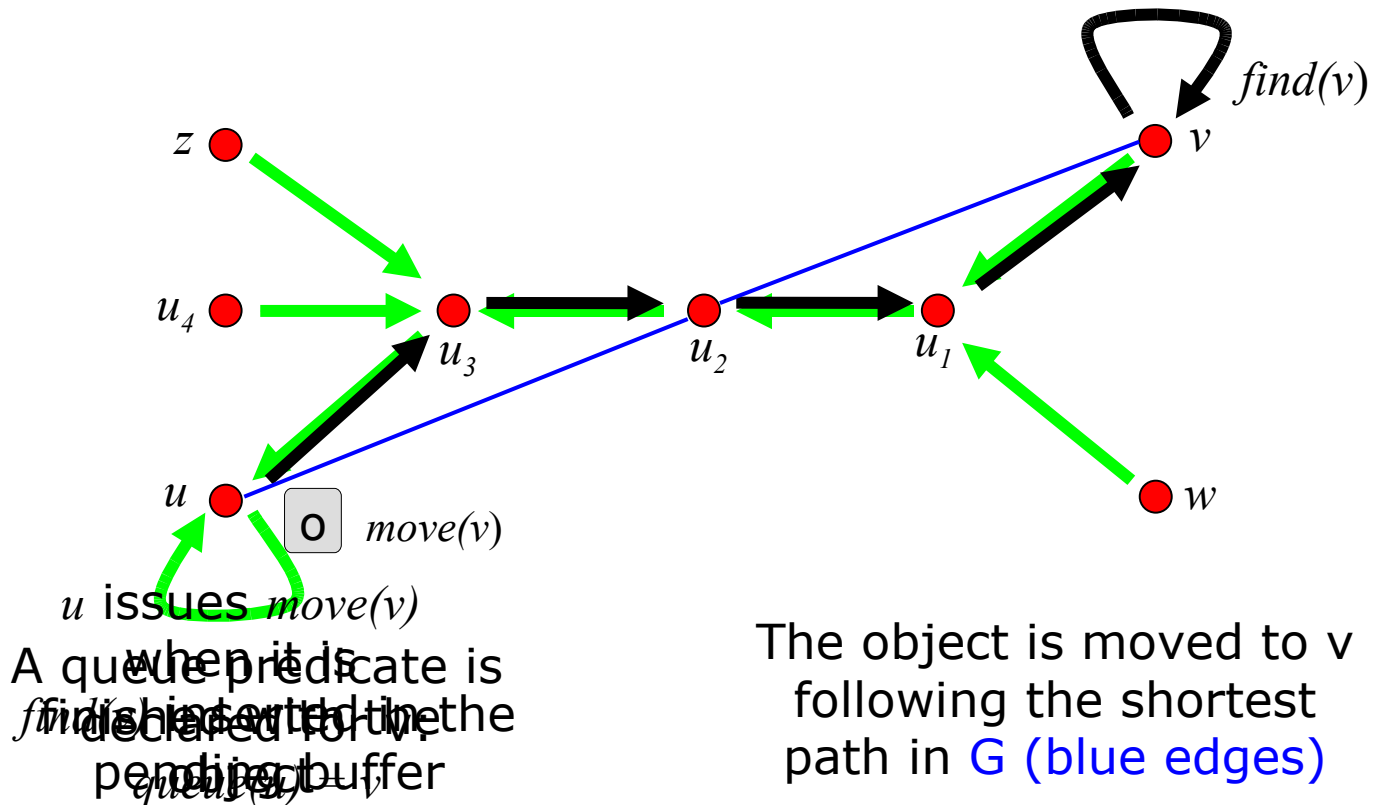


Messages and Constructs

- *link(u,v)*: Defines the spanning tree.
- *find(v)*: Request for the object issued by the node v .
- *move(v)*: The object is free to be moved to v . It travels with the object, following the links in the original graph.
- *pending(u,v)*: Every *link(u,v)* has a buffer that keeps the request. Not a FIFO, but reliable.
- *queue(u) = {v, NULL}*: A predicate attached with every node. Tells that u has to transfer the object to v when it is finished with the object.

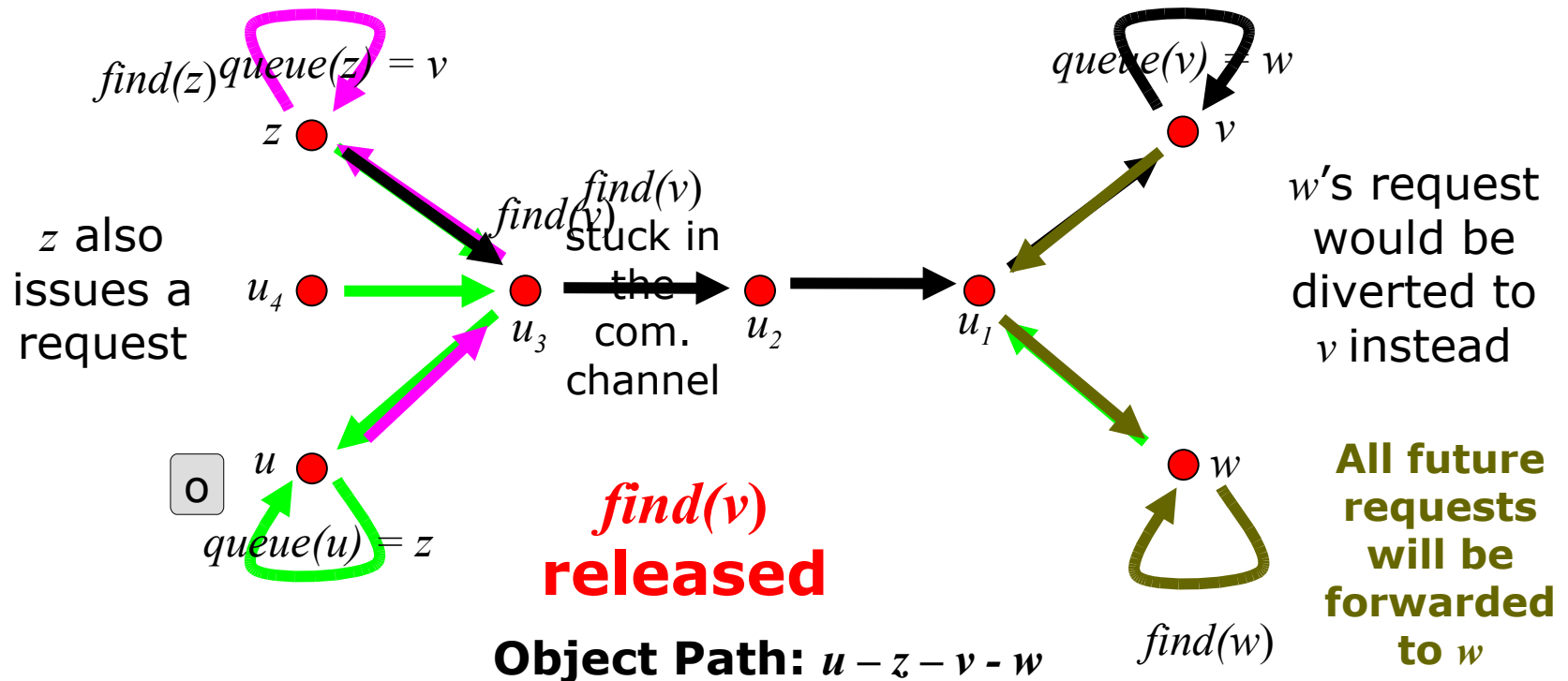
Working of the Protocol

- v issues a request $find(v)$ for the object.



Concurrent Requests

- $find(v)$ stuck in the communication channel.
- w also issues a request in the meanwhile.



Properties to Verify / Types of Goals

- **Graph Matching:** Can a particular node u be a terminal and all arrow paths end at u ?
- **Subgraph matching:** Can a particular node u be a terminal?
- **Graph isomorphism** Can an arbitrary node u_i be a terminal and all arrow paths end at u_i ?
- **Subgraph isomorphism:** Can an arbitrary node u_i be a terminal?

Cost Algebra (Edelkamp, Jabbar, Lluch-Lafuente AAAI-05)

A *cost algebra* is defined as a 6-tuple $\langle A, \sqcup, \times, \preceq, \mathbf{0}, \mathbf{1} \rangle$, such that

- (1) $\langle A, \times, \mathbf{1} \rangle$ is a **monoid**
- (2) \preceq is a **total order** induced by \sqcup
- (3) $\mathbf{0} = \bigsqcap A$ (**maximum**) and $\mathbf{1} = \bigsqcup A$ (**minimum**)
- (4) A is isotone

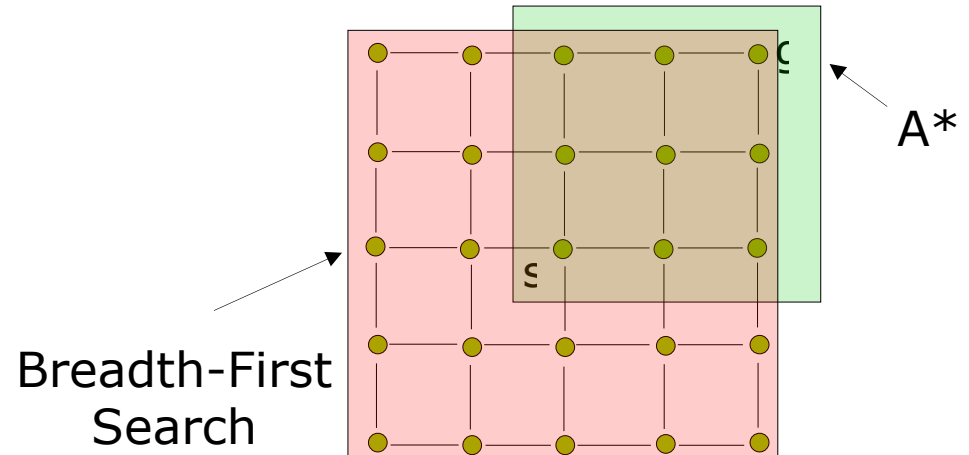
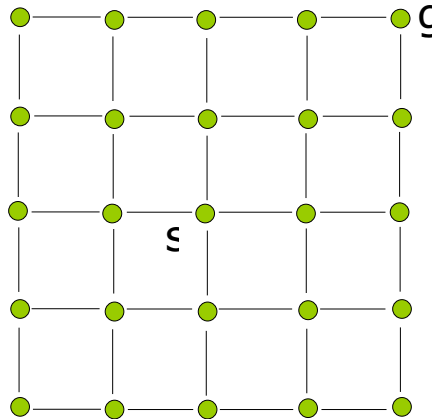
Optimization: $\langle \mathbb{R}^+ \cup \{+\infty\}, \min, +, \leq, +\infty, 0 \rangle$

Price, propagation delay.

Max/Min: $\langle \mathbb{R}^+ \cup \{+\infty\}, \max, \min, \geq, 0, +\infty \rangle$

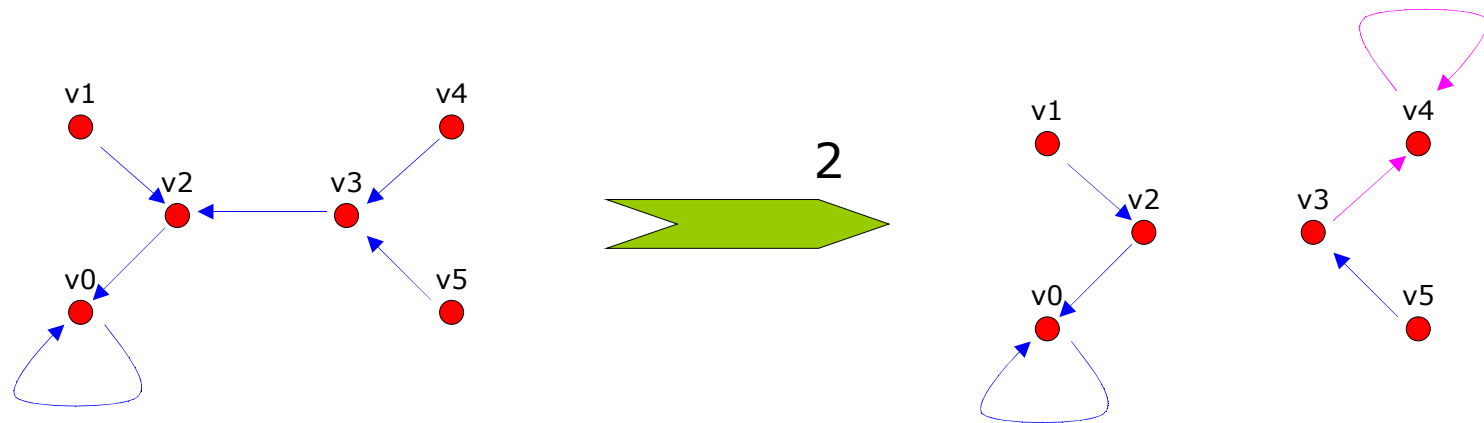
Bandwidth.

Heuristic Search



- ❑ **Admissible Heuristic:** Never over-estimates the optimal path.
Guarantees the optimal path.
- ❑ **Consistent Heuristic:** Never drops faster than the edge weight.
Guarantees the minimum number of expanded nodes.

Heuristics: Items to remove and insert

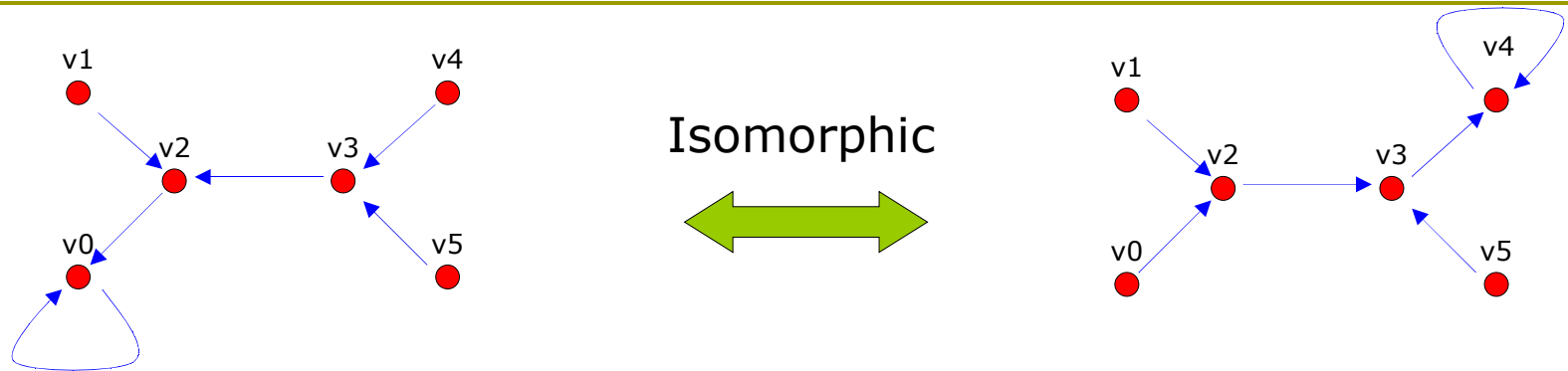


Since each transition can remove or add at most one edge, at least two transitions are necessary!



Consistent and admissible

Heuristic: Isomorphism Heuristic



Necessary conditions for isomorphism:

- Same **number** of nodes and edges
- One-to-one** correspondence between the in/out-degree of nodes.

Heuristic:

- For the current state and the goal graph, sort the list of nodes with increasing order of degrees.
- Compute the difference between the two lists.

Inconsistent and Inadmissible

Heuristic: Formula-Based Heuristic

- Goal graph is transformed to a closed negation-free FO graph formula.
- **Heuristic Idea:** Each false predicate contributes to an increase to the value.

$$h(\mathbf{true}, s) = \mathbf{1}$$

$$h(\mathbf{false}, s) = C$$

$$h(f \wedge g, s) = h(f, s) \times h(g, s)$$

$$h(f \vee g, s) = h(f, s) \sqcup h(g, s)$$

.....

min

**Inconsistent and
Inadmissible**

Heuristic: Hamming Distance

- Hamming distance between the binary state vector and the binary goal state vector.

State

10010100010001

Transition

State

10000100010000

Goal

10000100011000



Goal

10000100011000

Single transition can change multiple bits

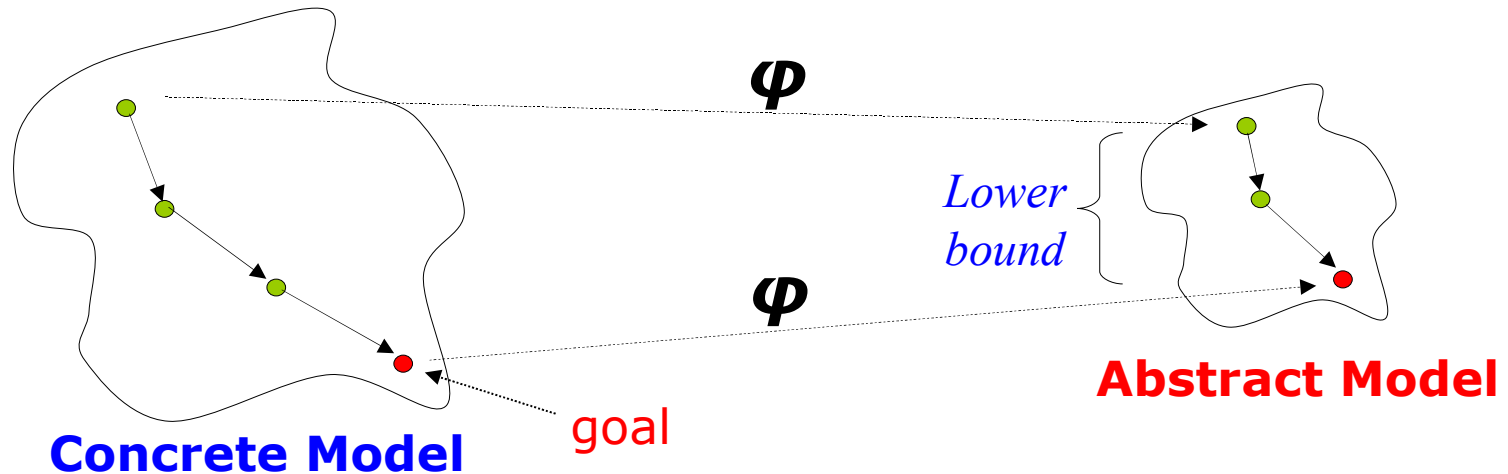


Inconsistent and Inadmissible Heuristic

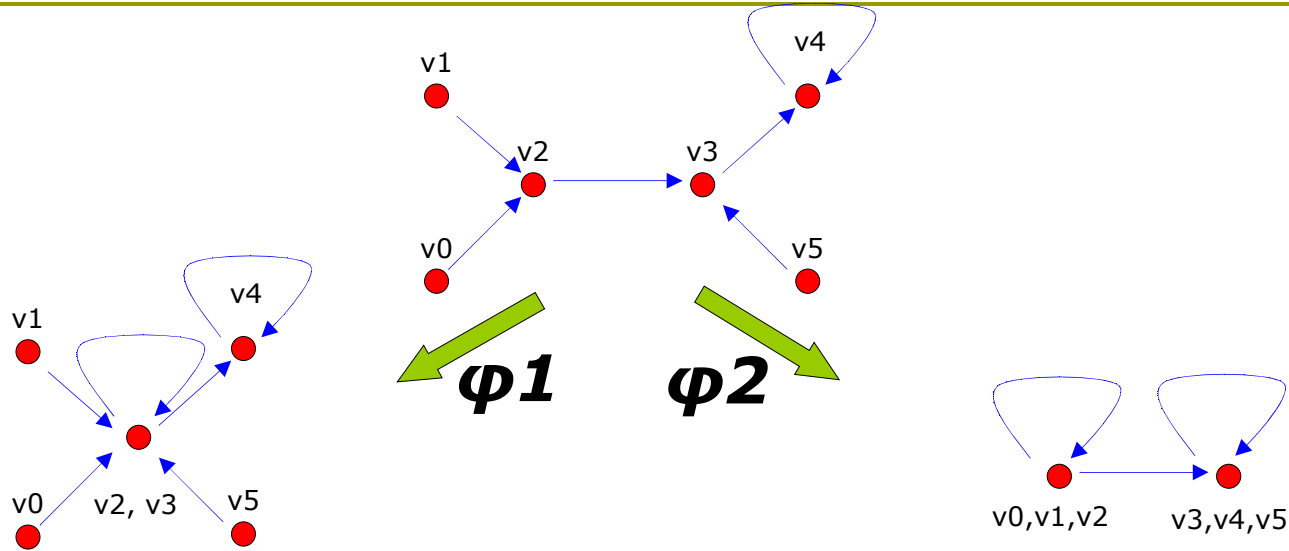
Since **state is a graph**, the bit vector can be computed by a **depth-first** or a **breadth-first** traversal.

Abstraction

- **Abstraction** can provide a significantly smaller system to verify.
- **Abstraction Databases** a.k.a. Pattern databases:
 - Make a **complete traversal** of the abstract model and **save the optimal path costs**.
 - Use these **costs** as **heuristic estimates**.



Abstraction in GTS



- **Multiple Abstraction Databases** a.k.a. Pattern databases:
 - Have to be **disjoint**
 - Can take a **MAX** for a **better heuristic**

Consistent and admissible

Experimental Evaluation

- ❑ Evaluated the approach on **Arrow Distributed Protocol** modeled as **Promela** – the input language of SPIN.
- ❑ Implemented on **HSF-SPIN** – extension of SPIN with heuristic search.
- ❑ Checked for **Property 2**:
 - Can a certain node v be a terminal and no other requests are queued over the network ?
- ❑ Three network **topologies**: **star, chain and tree**.
- ❑ **10 nodes** in each.

Reachability Experiments for Arrow Distributed Directory Protocol

	Dijkstra	Dept-First	Item to remove and insert	Hamming distance	Formula- based
star	DJK	DFS	$BF+h_n^1$	$BF+h_h$	$BF+h_f$
expanded nodes	38,701	6,253	30	6,334	30
solution cost	20	134	58	32	58
chain	DJK	DFS	$BF+h_n^1$	$BF+h_h$	$BF+h_f$
expanded nodes	413,466	78,112	38	1,49	38
solution cost	28	118	74	74	74
tree	DJK	DFS	$BF+h_n^1$	$BF+h_h$	$BF+h_f$
expanded nodes	126,579	24,875	34	24,727	34
solution cost	24	126	66	44	66

Optimality Experiments for Arrow Distributed Directory Protocol

	Dijkstra	Hamming + DFS goal state vector	Hamming + BFS goal state vector	Item to remove and insert	Hamming distance	Formula- based
star	DJK	$A^* + H_h(s_d)$	$A^* + H_h(s_b)$	$A^* + h_n^1$	$A^* + h_h$	$A^* + h_f$
expanded nodes	38,701	o.m.	1,255	13,447	117	206
solution cost	20	o.m.	20	20	20	20
chain	DJK	$A^* + H_h(s_d)$	$A^* + H_h(s_b)$	$A^* + h_n^1$	$A^* + h_h$	$A^* + h_f$
expanded nodes	413,466	26,622	1,245	106,629	1,620	198
solution cost	28	42	28	28	28	28
tree	DJK	$A^* + H_h(s_d)$	$A^* + H_h(s_b)$	$A^* + h_n^1$	$A^* + h_h$	$A^* + h_f$
expanded nodes	126,579	o.m.	1,481	33,720	6,197	224
solution cost	24	o.m.	24	24	24	24

Summary

- **Analysis** of GTS is a Hard.
- **Directed Model Checking** can help in reaching a goal faster.
- Presented a **portfolio of heuristics** for different kinds of goals.
- **Successfully modeled** Arrow Distributed Directory Protocol in Promela.
- Experimental Results with **HSF-SPIN** are very promising.
- Still some **limitations** in modeling the full-fledged specification of GTS.
 - **Dynamic** insertions and deletions of nodes and edges.
 - Can be circumvented to some extent by using a **pool of available objects**.