Real-Time Heuristic Search

Richard E. Korf

Van Do

Nov 3rd, 2016

Outline

- Motivation: Single Agent Search

- Real-time Single Agent Search

A* and Iterative-Deepening-A*

- A*: BFS $\mathrm{f}(n) = g(n) + h(n)$
- IDA*: DFS until $\mathrm{f(n)} > threshold$
- Exponential time to run
- Search entire space before first move

Real-time single-agent search

Apply assumptions of two-player games:

- Limited search horizon
- Commitment to move in constant time

Real-time single-agent search

1. Make individual move decisions

2. Find a solution

3. Learn from solving multiple trials

1. Make individual move decisions

Individual move decisions – essentially equivalent

- Minimin with alpha pruning
- Time-limited-A*
- Threshold-limited-IDA*

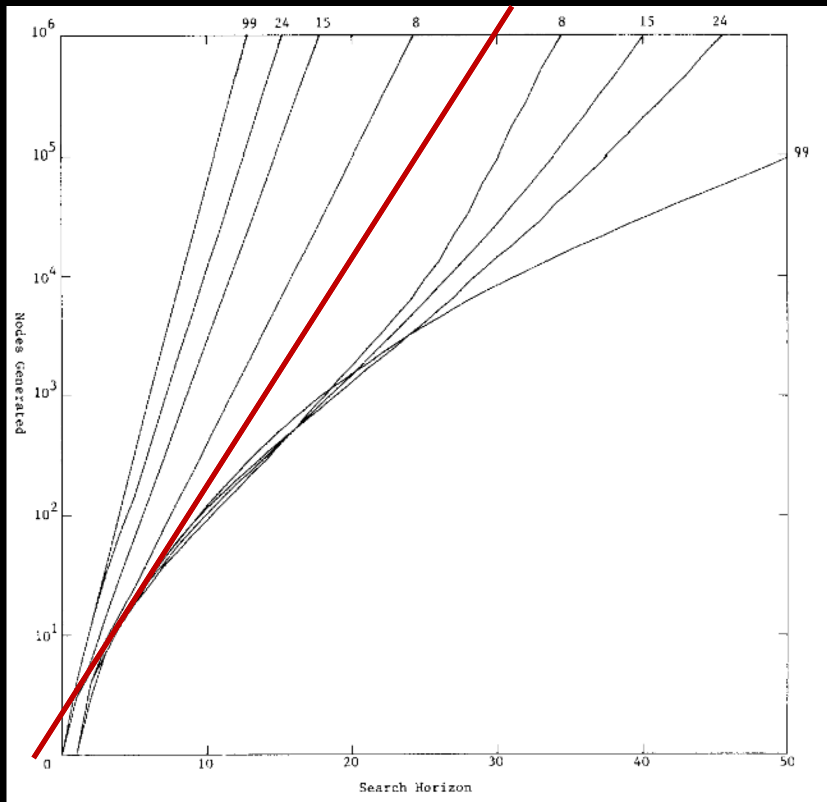
Minimin

- Search from current state to a fixed depth
- Calcualte heuristic function to nodes at the search frontier
- Single move is made in direction of best child (minimum value)

Alpha Pruning

- Monotonic cost function (equiv. Consistent heuristics): $f(child) \geq f(parent)$
- Keep $\alpha = lowest\ f\ value$
- Terminate search if $f(new\ node) \geq \alpha$

Search horizon w/ alpha pruning increases with increasing branching factor



Time-limited-A*

- Run A* until time runs out
- Make a move in the direction of the best node on OPEN node
- Exponential memory requirement

Threshold-limited-IDA*

- Run IDA* with $threshold \geq f(current\ state)$
- Make a move in the direction of min h value.

Individual move decisions – essentially equivalent

- Minimin with alpha pruning
- Time-limited-A*
- Threshold-limited-IDA*

2. Find a solution from individual move decisions

Find a solution from individual move

Real-time-A* (RTA*):

- Completeness
- Correctness
- Solution quality

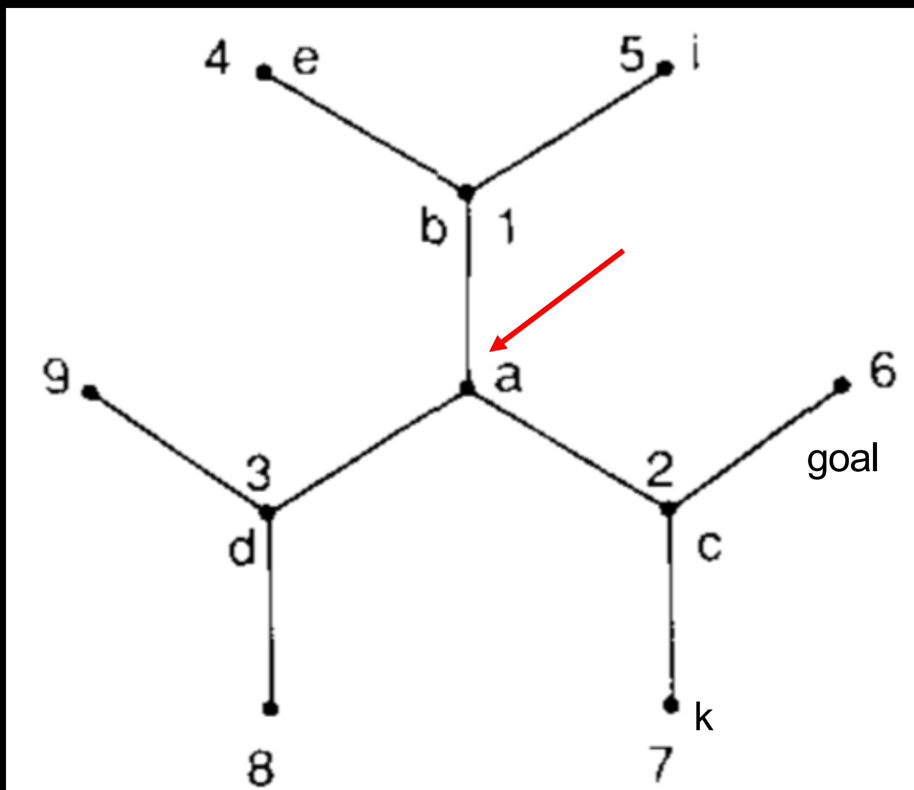
Real-time-A* (RTA*)

- BFS with

$$g(n) = distance\ from\ current\ state$$

- Make a move to node with min f value
- $h(current\ state) = \boldsymbol{second\ best}\ f\ value$

Real-time-A* (RTA*) – Example



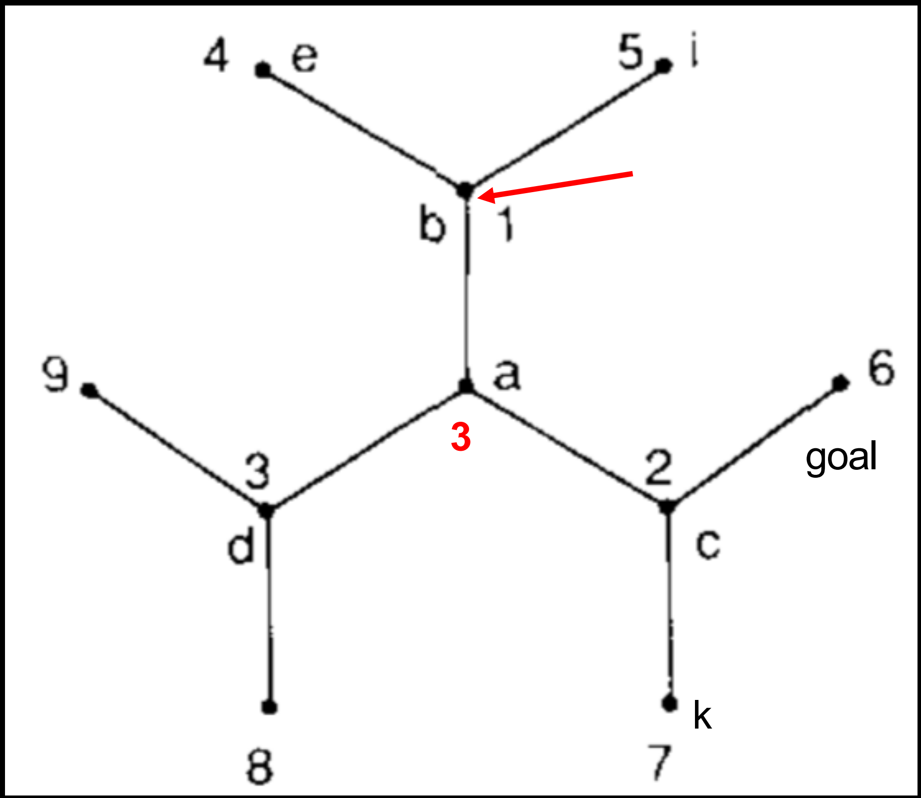
Start from a:

- $f(b) = 1 + 1 = 2$
- $f(c) = 1 + 2 = 3$
- $f(d) = 1 + 3 = 4$

Choose b

Update $h(a) = 3$ (f(c))

Real-time-A* (RTA*) - Example



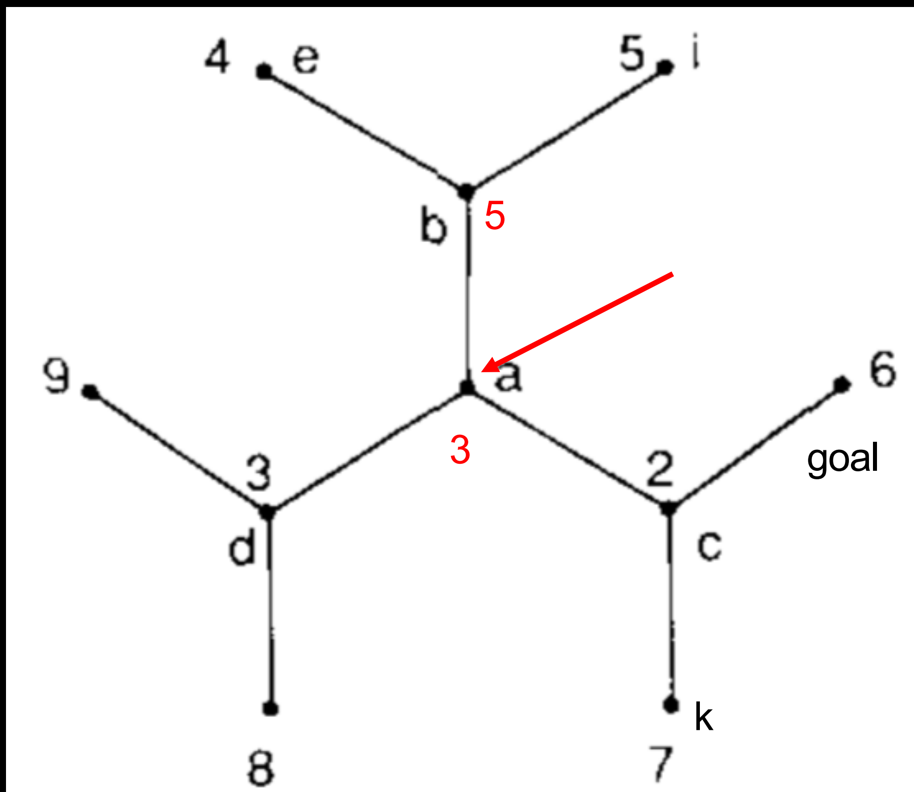
From node b:

- $f(e) = 1 + 4 = 5$
- $f(i) = 1 + 5 = 6$
- $f(a) = 1 + 3 = 4$

Choose a

Update $h(b) = 5$ (f(e))

Real-time-A* (RTA*) - Example



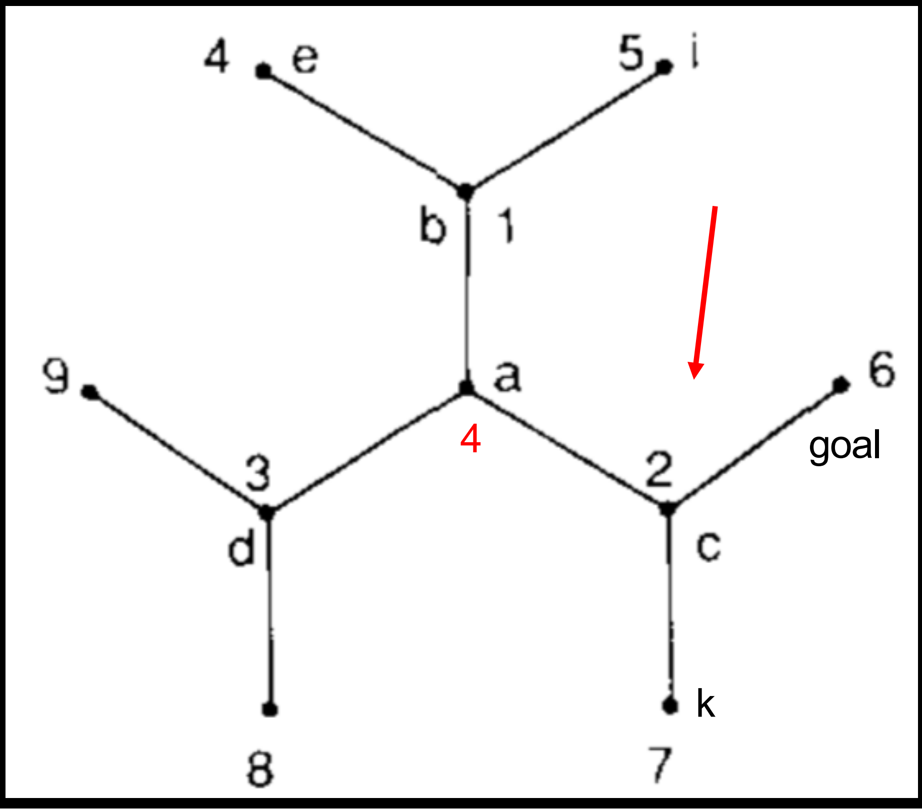
From node a:

- $f(b) = 1 + 5 = 6$
- $f(d) = 1 + 3 = 4$
- $f(c) = 1 + 2 = 3$

Choose c

Update $h(a) = 4$ (f(d))

Real-time-A* (RTA*) – Example – Not stuck in an infinite loop



From node c:

- $f(a) = 1 + 4 = 5$
- $f(k) = 1 + 7 = 8$
- $f(goal) = 1 + 6 = 7$

Goal found!

Not in an infinite loop

Completeness of RTA*

Theorem 1:

RTA* will find a solution when:

- Finite problem space
- Positive edge cost
- Finite heuristic function
- Goal state is reachable from every state

Correctness of RTA* - locally optimal decisions

Theorem 2:

RTA* will move along a path with:

- Estimated cost of reaching a goal is a minimum
- Based on cumulative search frontier at the time

Tie-breaking for RTA*

- Arbitrary tie-breaking: systemic bias
- Random tie-breaking
- Secondary search to resolve tie-breaking

Solution quality increases with search horizon

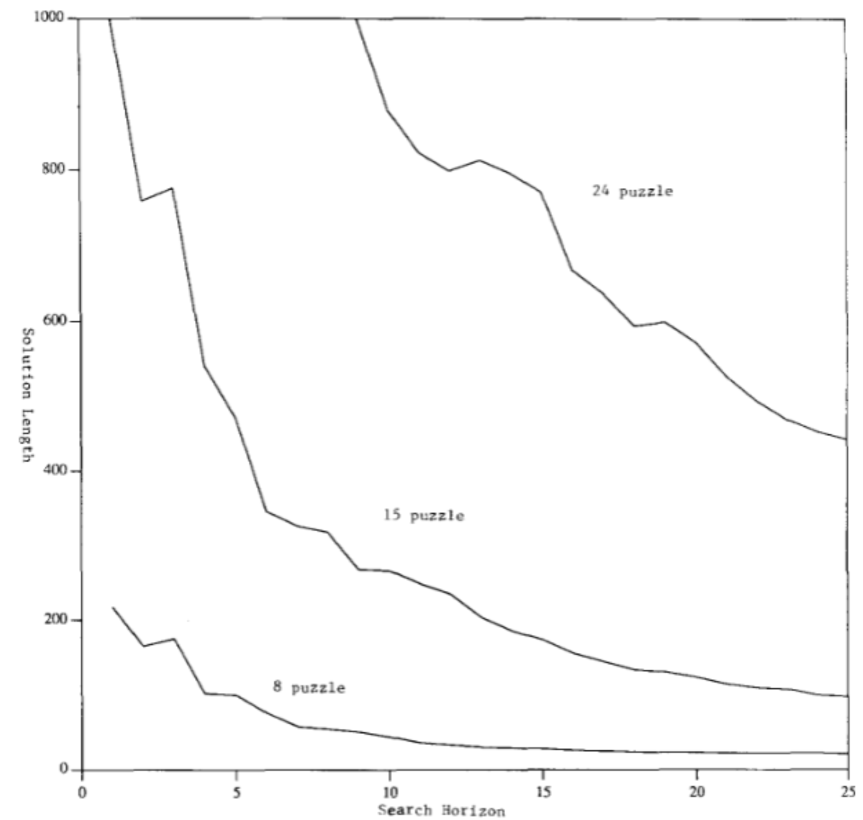


Fig. 4. Search horizon vs. solution length for RTA*.

- Solution quality: solution cost
- Search horizon: search depth

Computation vs. Execution

- Trade off between costs simulating vs. executing time
- Optimal search horizon is problem dependent

Learning from solving multiple trials

Learning from solving multiple trials

Learning RTA*

Convergence

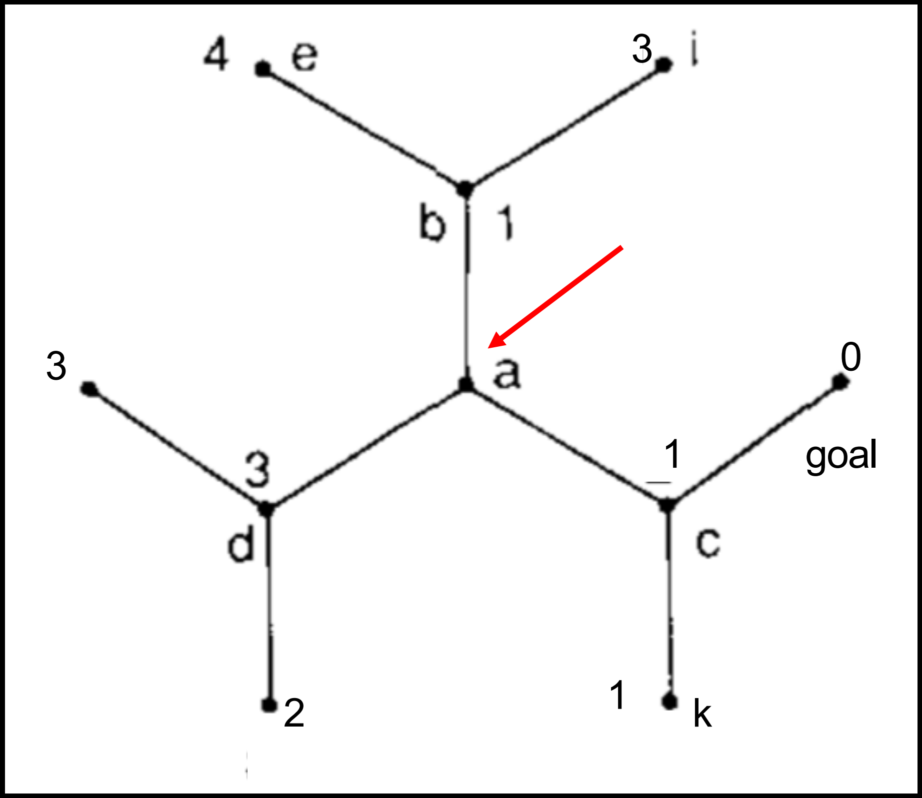
Learning RTA* (LRTA*)

- Infinite trials of LRTA*, each similar to RTA* except:

$$h(current\ state) = \boldsymbol{best\ f}\ value$$

- Store updated heuristics estimates of node from the previous run

LRTA* – Example – 1st trial



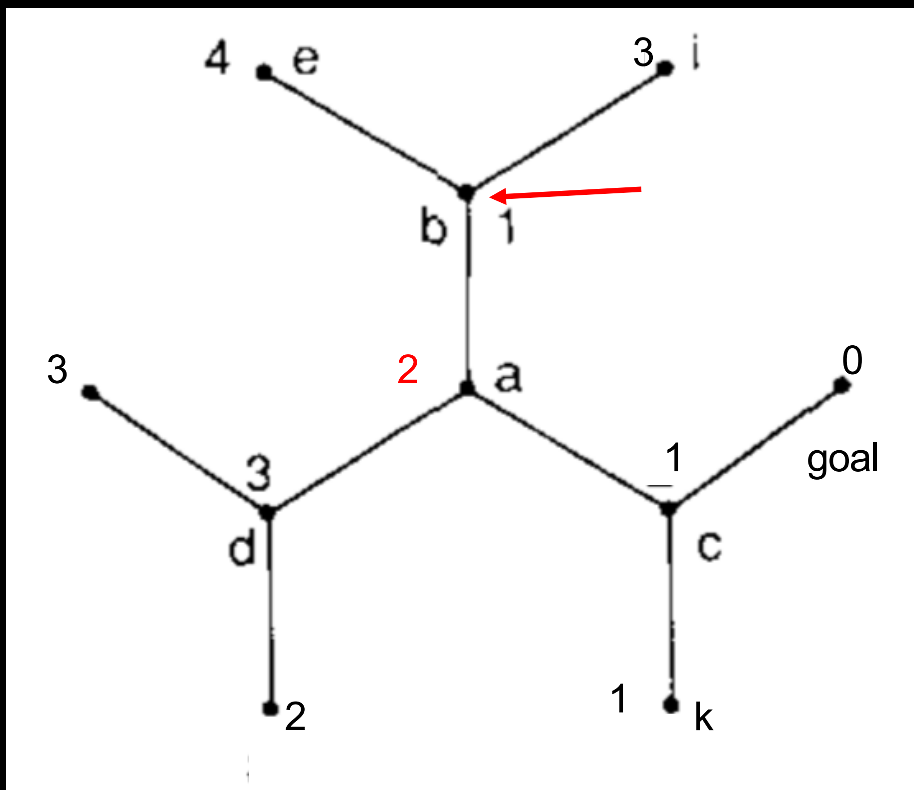
Start from a:

- $f(b) = 1 + 1 = 2$
- $f(c) = 1 + 1 = 2$
- $f(d) = 1 + 3 = 4$

Break tie randomly. Choose b

Update $h(a) = 2$ (f(b))

LRTA* – Example – 1st trial



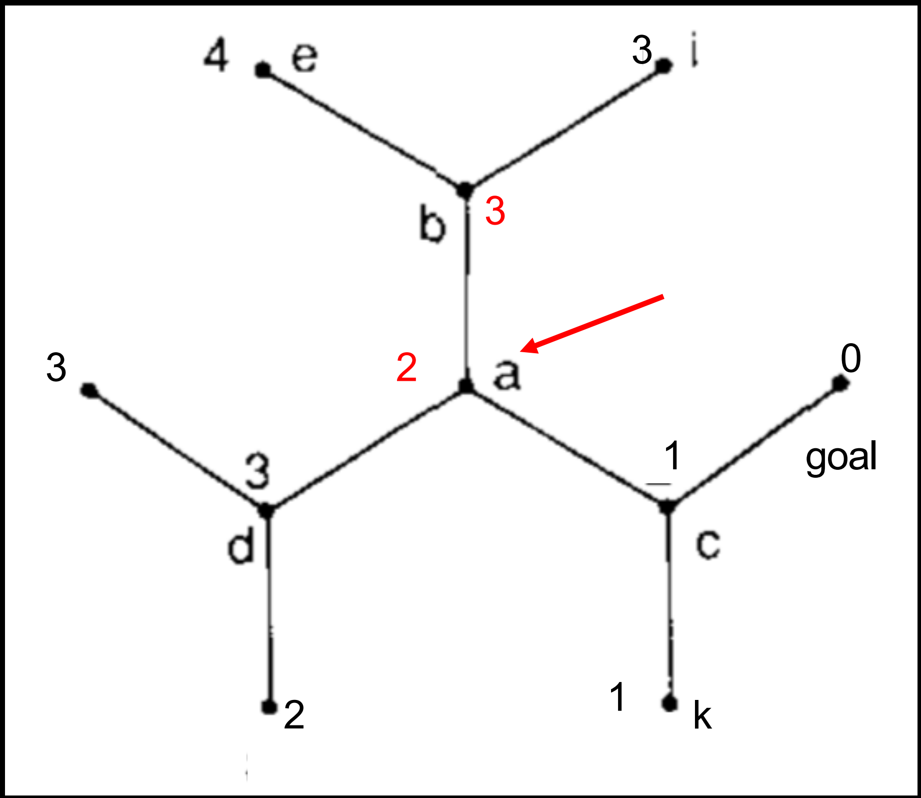
From b:

- $f(e) = 1 + 4 = 5$
- $f(i) = 1 + 3 = 4$
- $f(a) = 1 + 2 = 3$

Choose a

Update $h(b) = 3$ (f(a))

LRTA* – Example – 1st trial



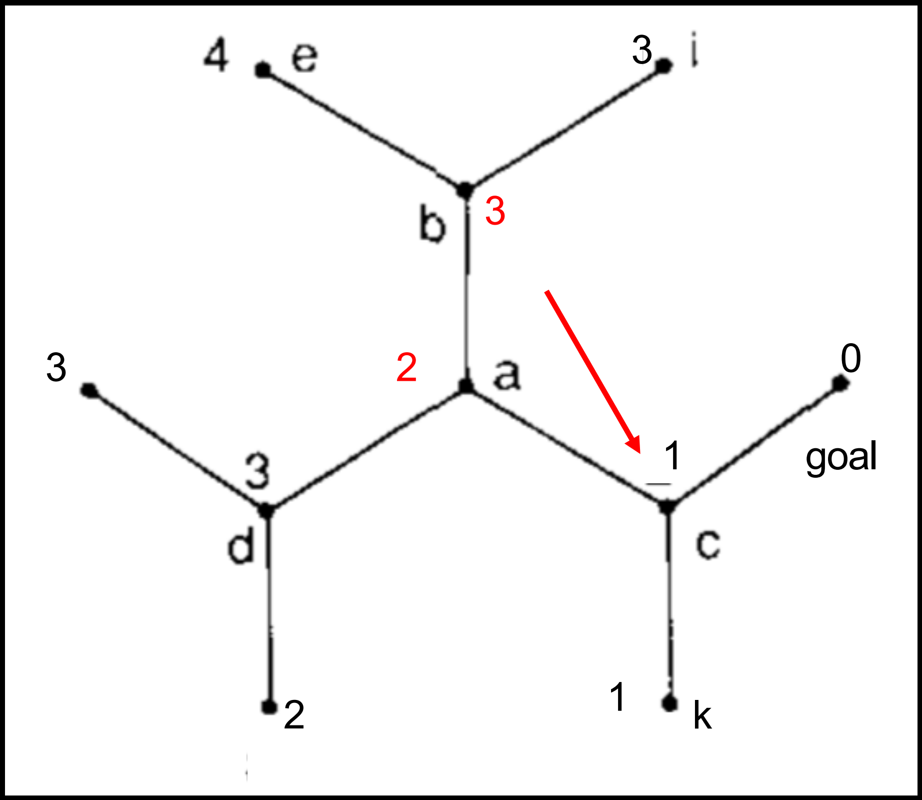
From a:

- $f(b) = 1 + 3 = 4$
- $f(c) = 1 + 1 = 2$
- $f(d) = 1 + 3 = 4$

Choose c

Update $h(a) = 2$ (f(c))

LRTA* – Example – 1st trial

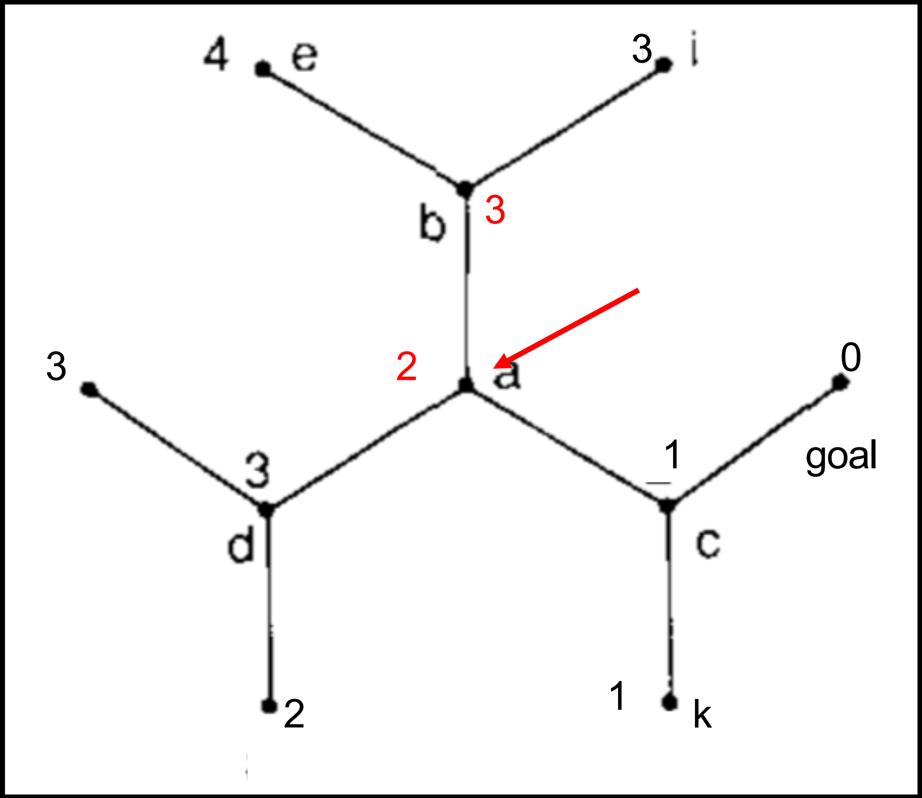


From c:

- $f(a) = 1 + 2 = 3$
- $f(k) = 1 + 1 = 2$
- $f(goal) = 1 + 0 = 2$

Goal found!

LRTA* – Example – 2nd trial



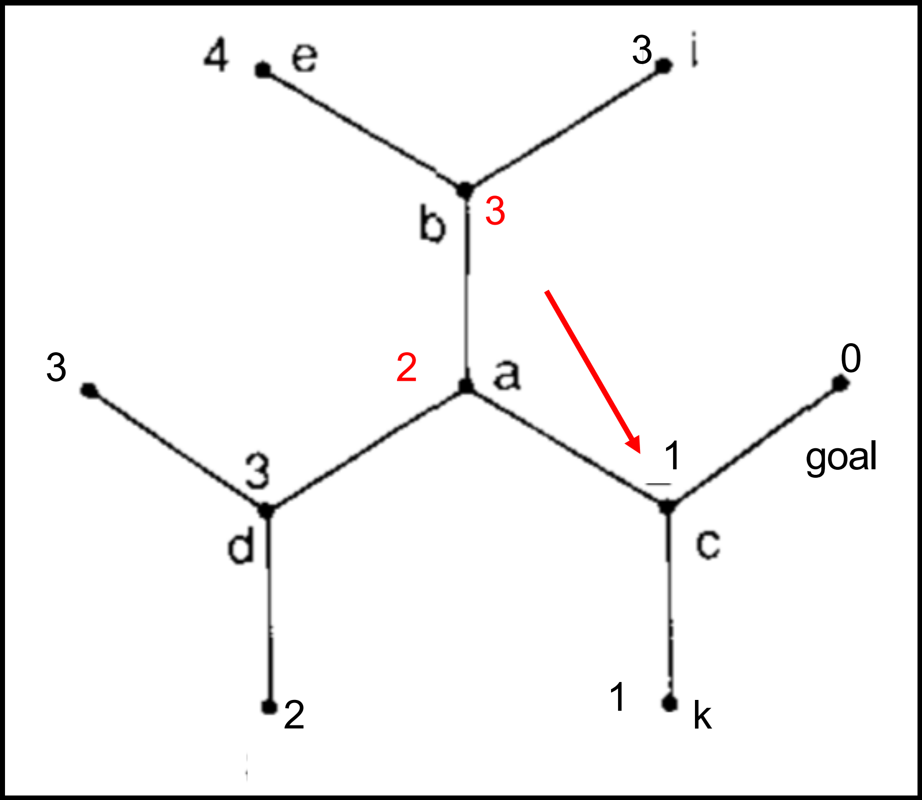
From a:

- $f(b) = 1 + 3 = 4$
- $f(c) = 1 + 1 = 2$
- $f(d) = 1 + 3 = 4$

Move to c.

Update $h(a) = 2$

LRTA* – Example – 2nd trial



From a:

- $f(a) = 1 + 2 = 3$
- $f(k) = 1 + 1 = 2$
- $f(goal) = 1 + 0 = 2$

Goal found!

LRTA*

- Retains completeness property of RTA*
- Not always make locally optimal decisions

Convergence of LRTA* after multiple trials

Theorem 3: heuristic values will converge to exact values along every optimal path

- Non-overestimating initial heuristic values
- Infinite repeated trials of LRTA*
- Finite problem space, positive edge costs

Intuition for convergence of LRTA*

- Value of a node will be corrected after visited by LRTA* if values of its successors are correct
- Working backwards from goal and do sequential correction for predecessor nodes.

Conclusions:

- Minimin with Alpha pruning
- RTA*
- LRTA*