Course code: BSINARI1KU

IAI Exam 2022

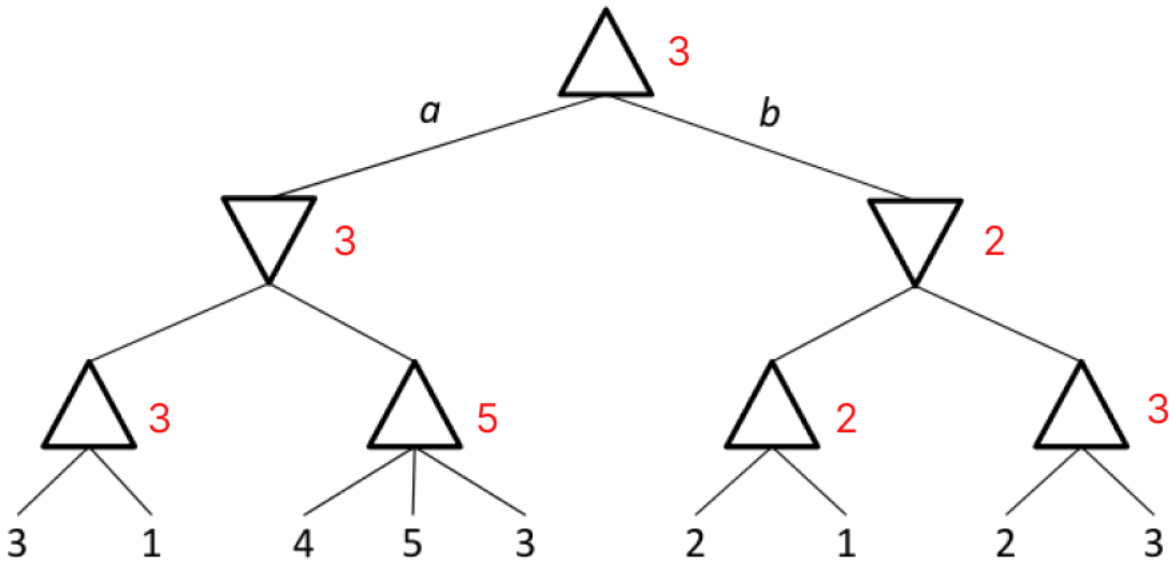
Kristoffer Højelse

krbh@itu.dk

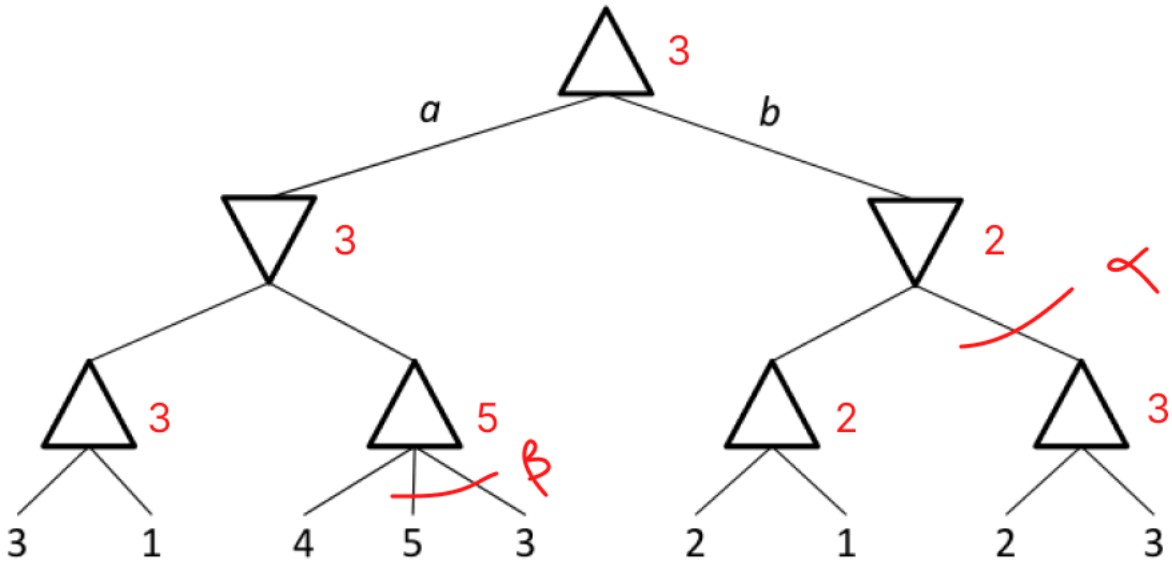
02 Juni 2022

1

1.1



1.2



1.3

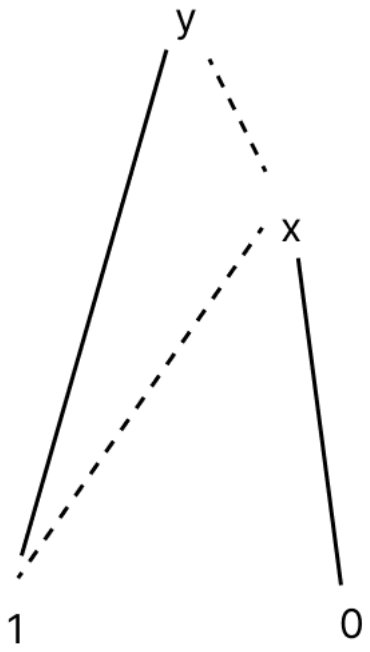
No both the α-cut and β-cut are done at the earliest possible points.

β-cut just after evaluating the first child (=4) of max, after one whole max node has been evaluated (=3).

α-cut just after evaluating the first child (=2) of min, after on whole min node has been evaluated (=3 at action a).

2

2.1



2.2

$(x \implies y) \wedge \neg x$

$\equiv (\neg x \vee y) \wedge \neg x$ [implication elimination]

$\equiv (\neg x \wedge \neg x) \vee (\neg x \wedge y)$ [distributivity of $\wedge$ over $\vee$]

$\equiv \neg x \vee (\neg x \wedge y)$ [simplify]

Truth table:

x	y	$\neg x \vee (\neg x \wedge y)$
0	0	1
0	1	1
1	0	0
1	1	0

$\equiv \neg x$

Turns out, the expression was not dependent on y.

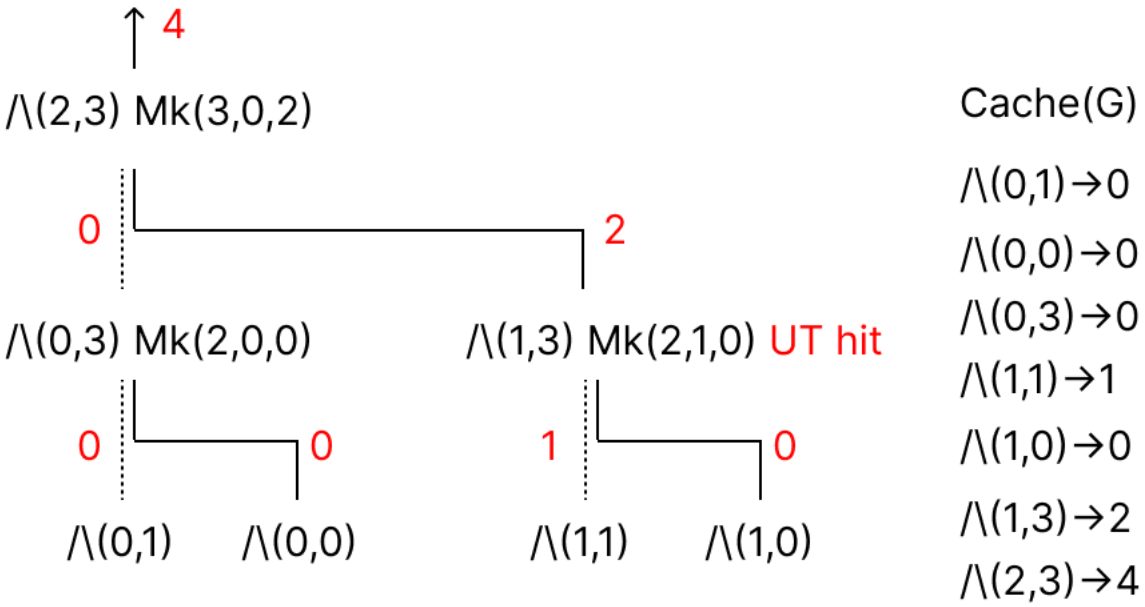
2.3

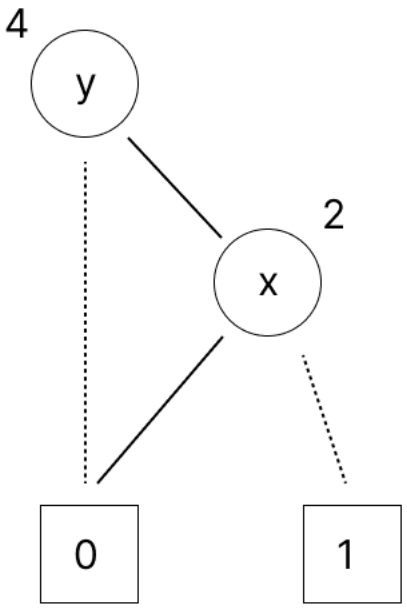
No.

Counter example:

$\neg x \wedge y$ satisfies the knowledge base.

2.4





2.5

The ROBDD is a contraposition of the knowledge base of the entails statement:

$(x \implies y) \wedge \neg x \equiv (\neg y \implies \neg x) \wedge \neg x$

contraposition rule: $(\alpha \implies \beta) \equiv (\neg\beta \implies \neg\alpha)$

3

3.1

Yes.

Given the SLD heuristic, we know that it must satisfy the triangle inequality (any path in the graph is longer than the Straight Line Distance), and is therefore an admissible heuristic.

3.2

Yes.

We need that $h(s') + c(s, a, s') \leq h(s)$ where h is the SLD heuristic, s is a node, and s' is a successor to node s, and $c(s, a, s')$ is the cost from s to s'.

The SLD heuristic satisfies the triangle inequality, which in turn satisfies the consistency constraint by definition.

3.3

iter	frontier (f, X, g, h)	reached (X,g)
0	(3, C, 0, 3)	(C,0)
1	(3, F, 1, 2), (5, D, 2, 3), (7, A, 2, 5)	(C,0), (F,1), (D,2), (A,2)
2	(4, G, 2, 2), (5, D, 2, 3), (7, A, 2, 5)	(C,0), (F,1), (D,2), (A,2), (G,2)
3	(5, L, 5, 0), (5, K, 3, 2), (5, H, 4, 1), (5, D, 2, 3), (7, A, 2, 5), (7, J, 4, 3)	(C,0), (F,1), (D,2), (A,2), (G,2), (L,5), (K,3), (H,4), (J,4)

The path: $C \to F \to G \to L$

3.4

It will be more like Greedy best-first search which means:

disadvantage: its a suboptimal search algorithm, not guaranteed to find the optimal solution.

advantage: its a bounded suboptimal search algorithm, giving a "good enough" answer. And it will find a valid path quicker (in many cases, not all cases).

3.5

Yes. This path is then cheaper: $C \to D \to H \to L$ costing 6, than the path $C \to F \to G \to L$ which then costs 7.

3.6

The directionallity by itself (into or outof the inner city) is no problem for A* as you can find many implementations supporting directional graphs.

Standard implementations of A* does not normally support action cost dependent on the path to get there.

It will no doubt require a lot of modifications, but I also can't figure out why you couldn't do this.

4

4.1

Not enough information for a definite answer: "variable" is ambiguous, might refer to programming variable or variables in a configuration problem like Container Stowage Problem(where locations are modelled as variables).

Assuming "variable" referes to variables in a configuration problem like Container Stowage Problem(where locations are modelled as variables):

No.

The actions to reach the neighborhood of a state can be modelled with swaps: effectively (re)assigning 2 or more variables per iteration.

4.2

Yes.

Since WalkSat flips variables randomly, and is therefore not guaranteed to explore the full state space and therefore is incomplete.

4.3

Yes.

It will explore every domain value ordering. So relevant effects like running time, will be independent on domain value ordering.

With the same Select-Unassigned-Variable() ordering, it is still not complete.

4.4

No.

AC-3 only checks arc-consistency. "Has at least 1 solution" is a improper subset of arc-consistency.

4.5

Yes.

Only point of contention: A* can traverse a graph finding the same point twice, effectively finding a cycle, and trees cannot have cycles. But will not add a cycle to the shortest-path-tree that it creates (in some implementation this tree is only an internal data-structure).