Paper Review: Multiagent Rollout Algorithms and Reinforcement Learning

Abdullah Mamun

April 8, 2022

About this paper

Multiagent Rollout Algorithms and Reinforcement Learning

Dimitri Bertsekas†

Abstract

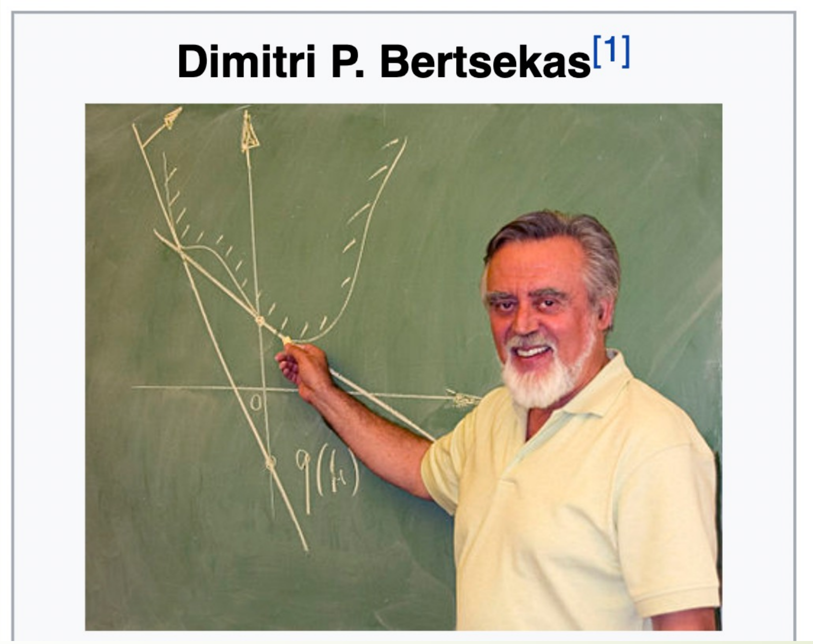
We consider finite and infinite horizon dynamic programming problems, where the control at each stage consists of several distinct decisions, each one made by one of several agents. We introduce an approach, whereby at every stage, each agent's decision is made by executing a local rollout algorithm that uses a base policy, together with some coordinating information from the other agents. The amount of local computation required at every stage by each agent is independent of the number of agents, while the amount of total computation (over all agents) grows linearly with the number of agents. By contrast, with the standard rollout algorithm, the amount of total computation grows exponentially with the number of agents. Despite the drastic reduction in required computation, we show that our algorithm has the fundamental cost improvement property of rollout: an improved performance relative to the base policy. We also discuss possibilities to improve further the method's computational efficiency through limited agent coordination and parallelization of the agents' computations. Finally, we explore related approximate policy iteration algorithms for infinite horizon problems, and we prove that the cost improvement property steers the algorithm towards convergence to an agent-by-agent optimal policy.

About this paper

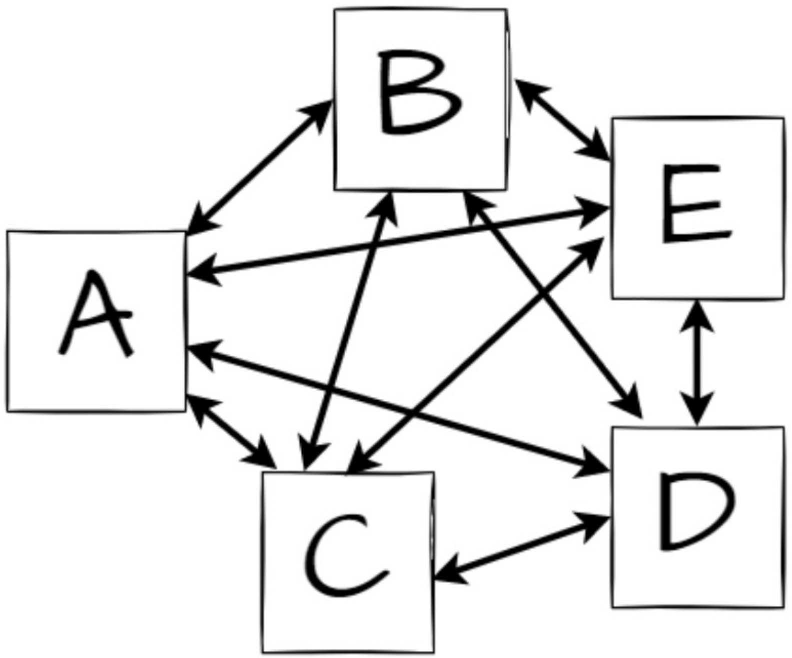
- Published on arxiv in 2019.
- Proposes a modified rollout algorithm for multi-agent systems named Multiagent Rollout
- Cited by 19 as of Apr 8, 2022

About the author

- Professor at MIT and ASU
- Total citations to date: 117, 236
- h-index: 101



Example problem 1: Traveling Salesman

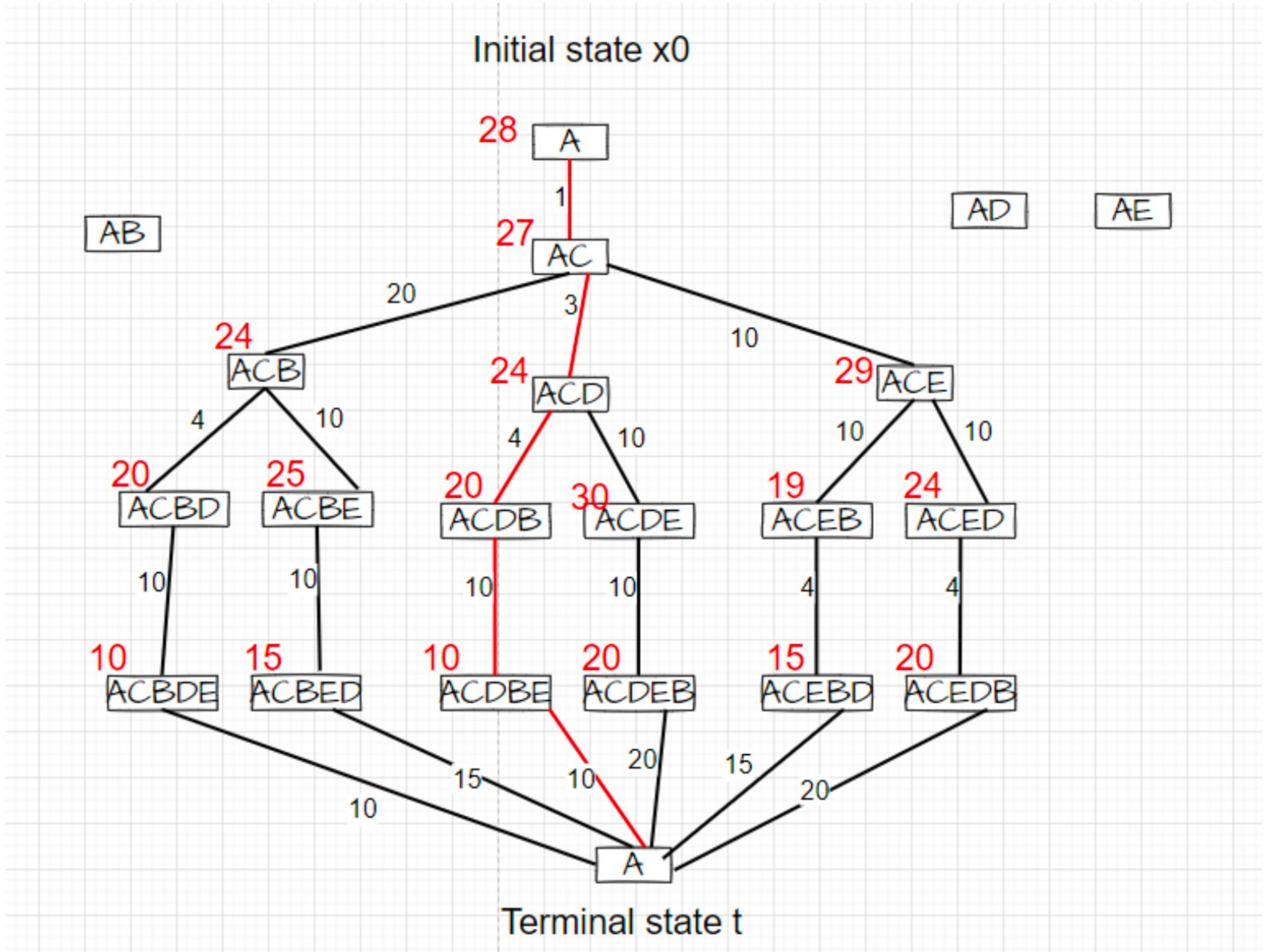


Five cities

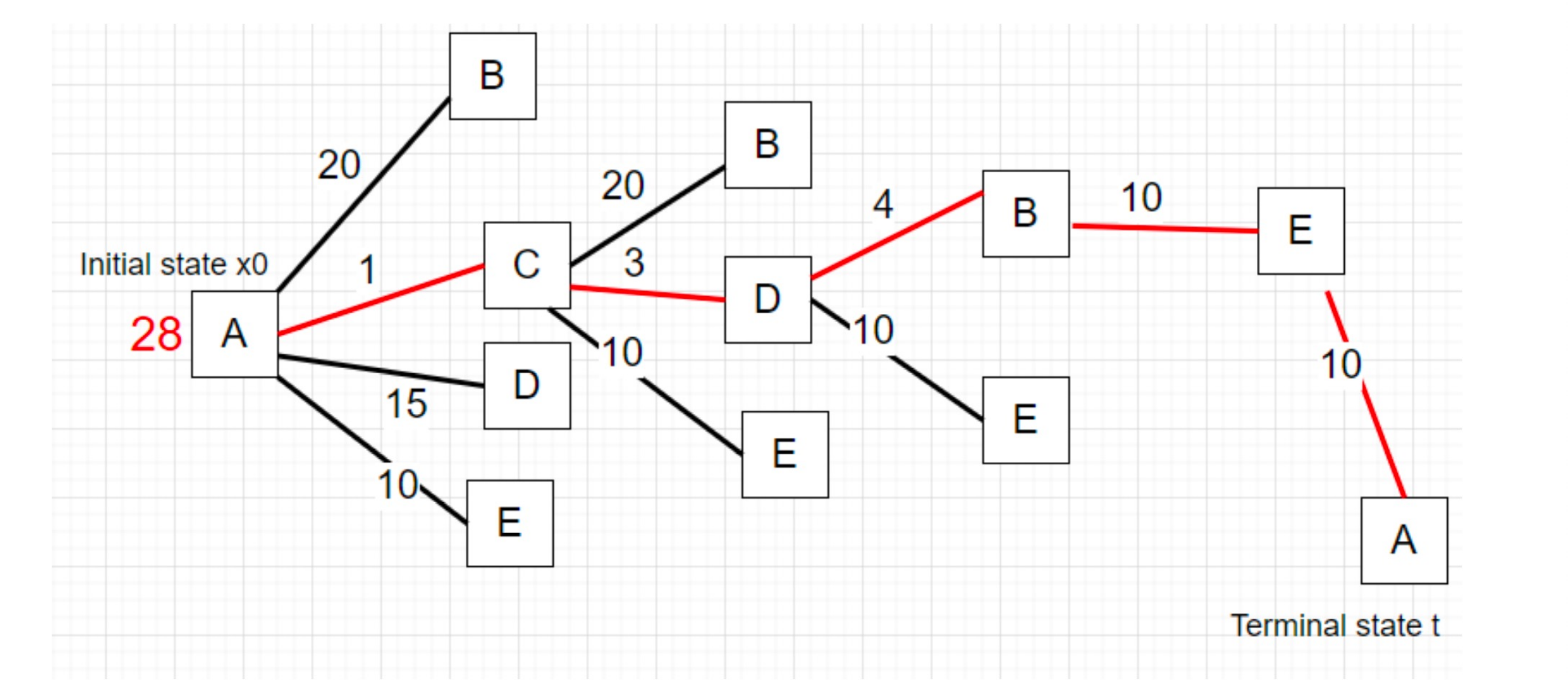
Matrix of Intercity Travel Costs				
	5	1	15	10
20		20	4	10
1	20		3	10
15	4	3		10
10	10	10	10	

Solution 1: Brute-force O (n!)

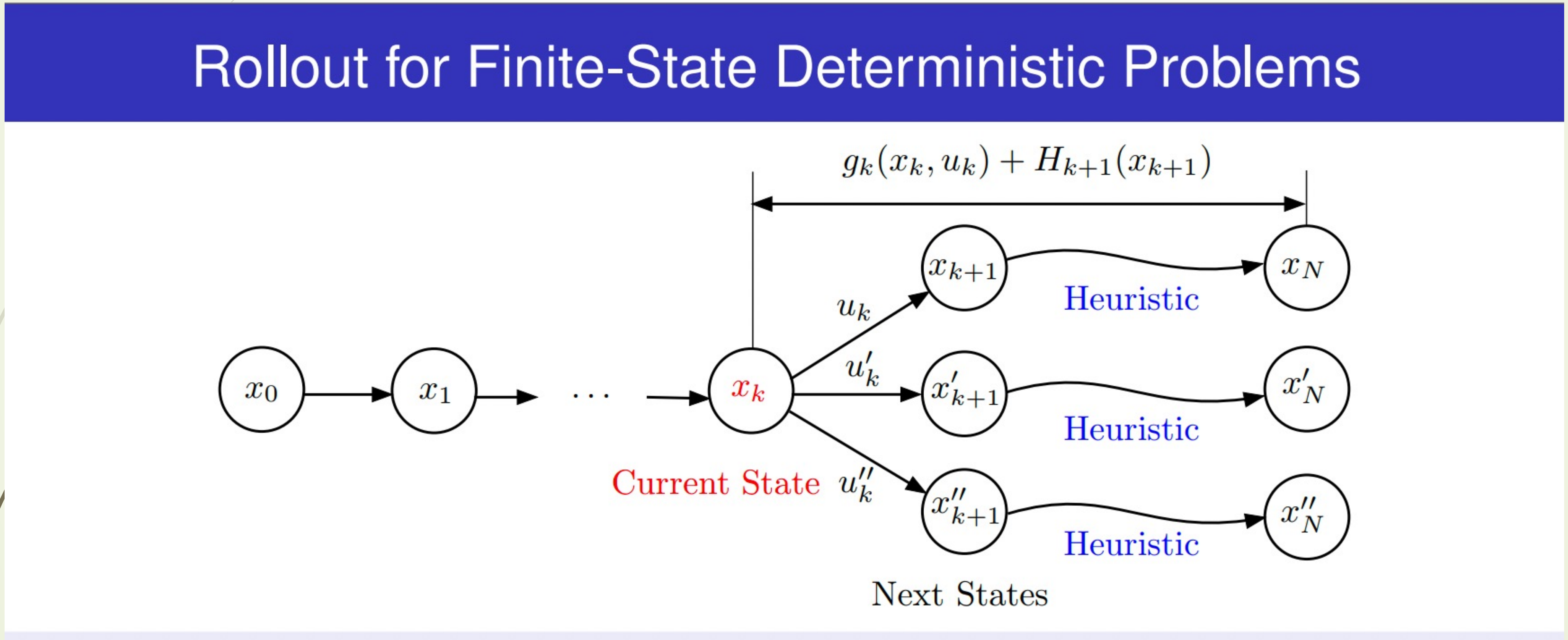
If the 2nd city to visit is C, then the optimal tour is: ACDBEA and the cost is 28



Solution 2: Heuristic (e.g. Nearest neighbor) $O(n^2)$

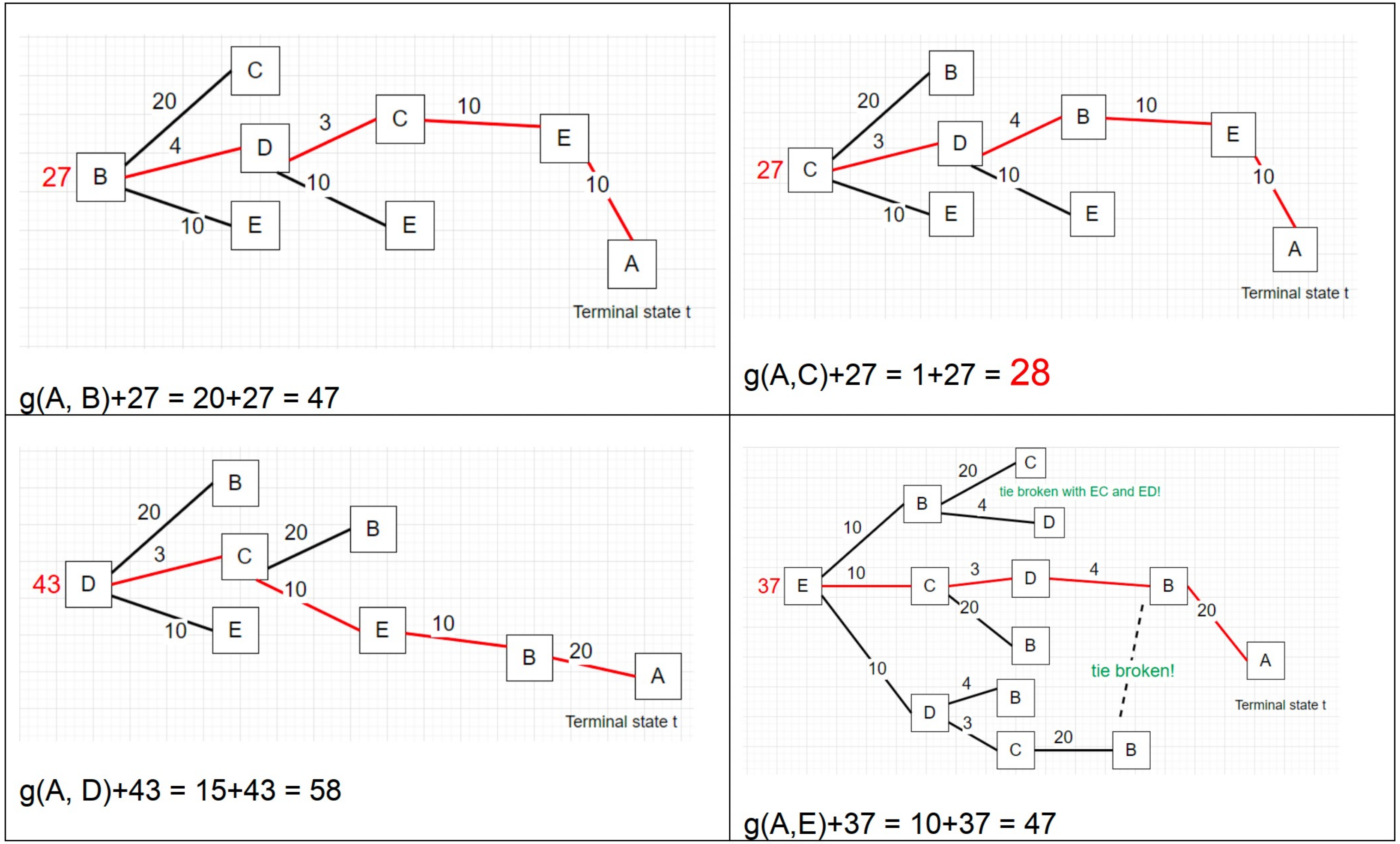


Solution 3: Rollout (with a base heuristic)



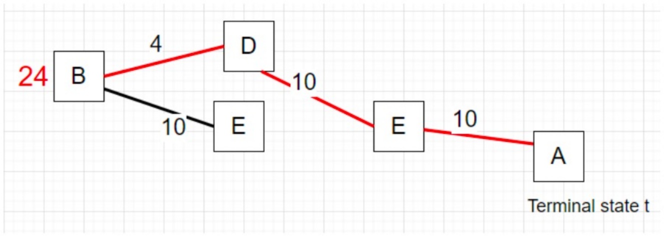
Source: Lecture slides by Prof. Dimitri Bertsekas, http://web.mit.edu/dimitrib/www/RLbook.html, last accessed 4/13/2022

Solution 3: Rollout (with the nearest-neighbor huristic)
Finding the 2nd city

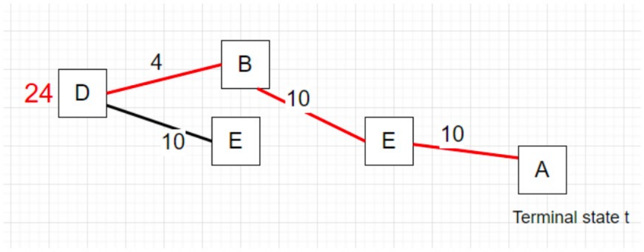


Solution 3: Rollout (with the nearest-neighbor huristic)

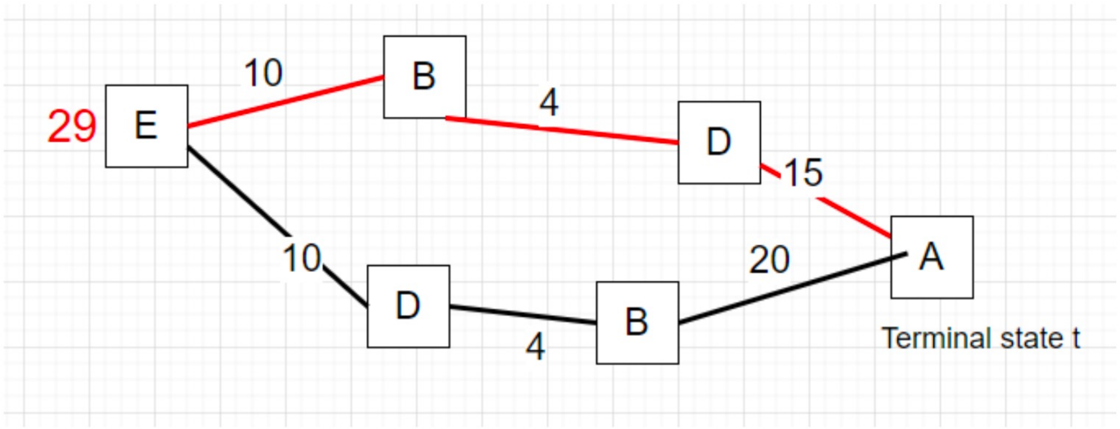
Finding the 3rd city



g(A,C) + g(C, B) + 24 = 1+ 20+24 = 44



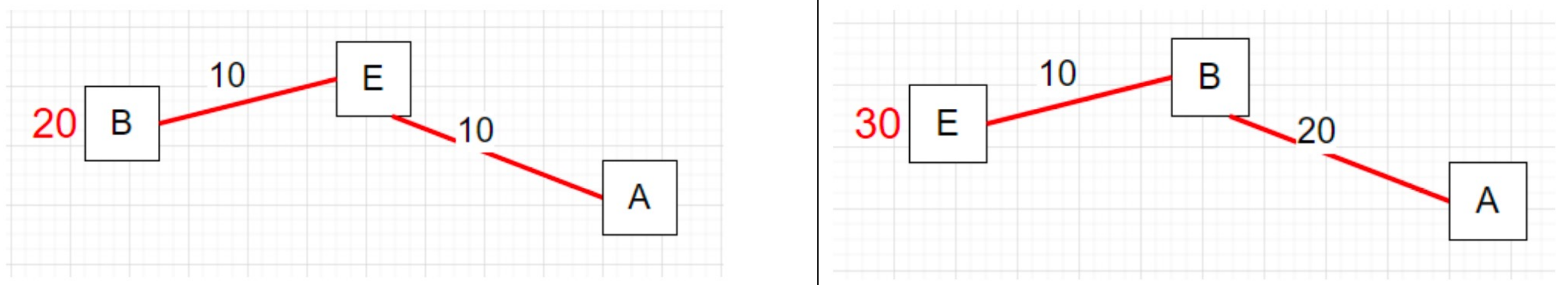
g(A,C) + g(C, D) + 24 = 1+3+24 = 28



g(A,C) + g(C, E) + 24 = 1+10+29 = 40

Solution 3: Rollout (with the nearest-neighbor huristic) – O (n^3)

Finding the 4th city



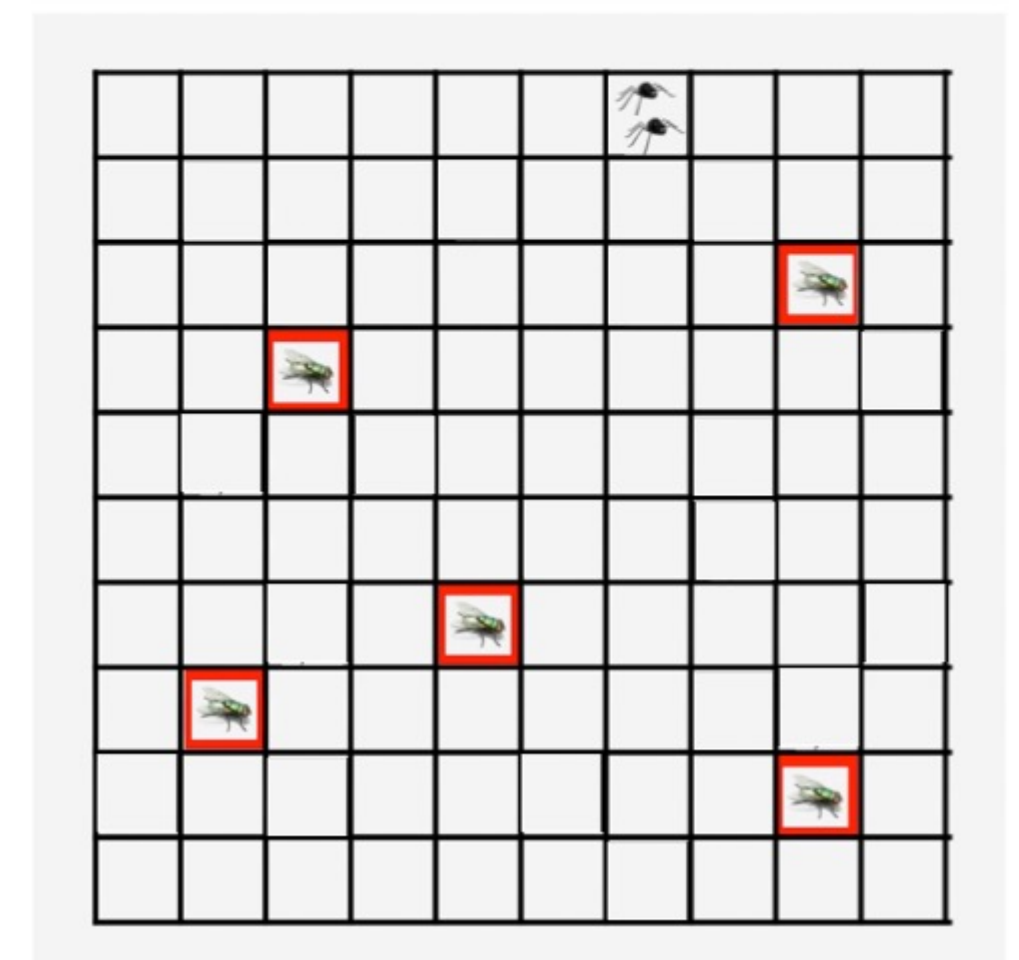
g(A,C)+g(C,D)+g(D,B)+20 = 1 + 3 + 4 + 20 = 28	g(A,C)+g(C,D)+g(D,E)+30 = 1 + 3 + 10 + 30 = 44

So, from the cases, we can say that B must be the fourth city and E is the fifth one.

So, the approximate optimal tour according to the rollout method is: ACDBEA

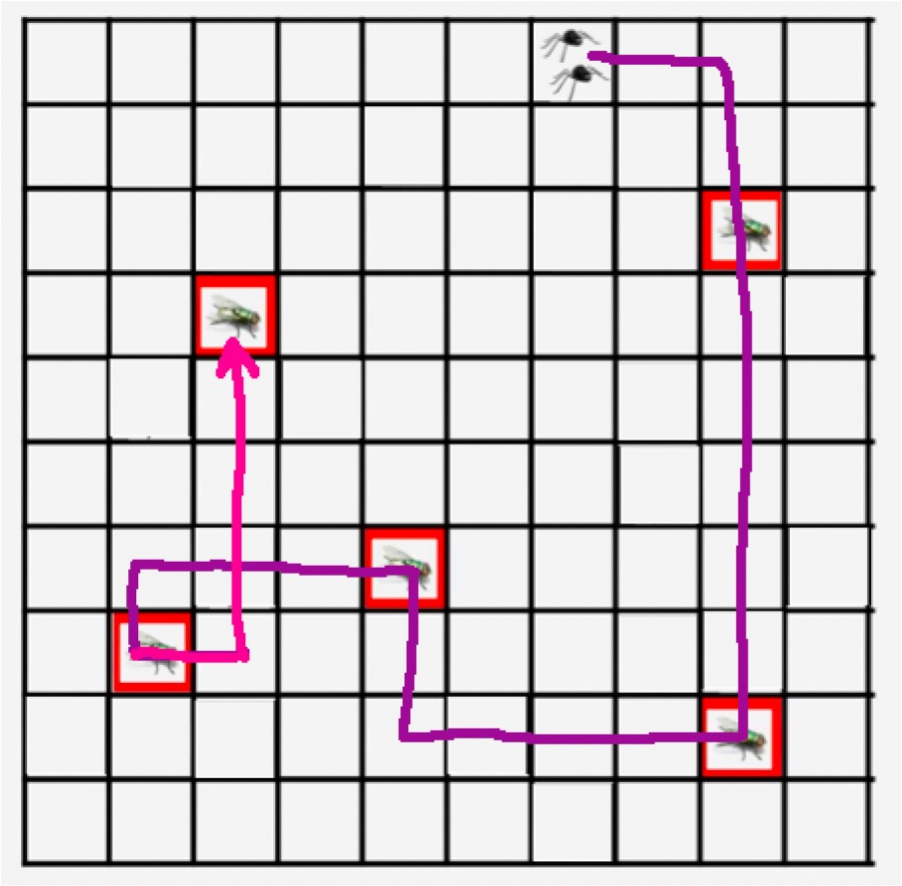
and the cost is: 28.

Problem of Rollout? – Still can be very expensive especially in a multi-agent situation



For N agents with M actions each. The number of possible combinations of actions is $O(N^M)$.

Base heuristic – nearest fly



25 steps to reach the goal.

Multiagent vs Standard rollout:

Standard rollout:
All agents act as a single unified system together.

Multiagent: O (N*M) instead of $O(N^M)$

Agent 1 takes action according to only its possible actions and thinking that others will follow base heuristic. Then it broadcasts its action

Agent 2 takes action according to its possible actions with rollout but also takes account the previous action

And so on for the next agents.

Multiagent

(c) Multiagent rollout

Here, in Example 1.6.4, it was clear that each spider will assume that the other spiders will move towards their closest surviving spiders. <u>Each spider will broadcast</u>

<u>its move to other spiders, but, what I understood is that, it will not change any of the other spiders' moves unless this spider has killed a fly just now.</u>

Results – standard rollout and multiagent rollout

Summary of results

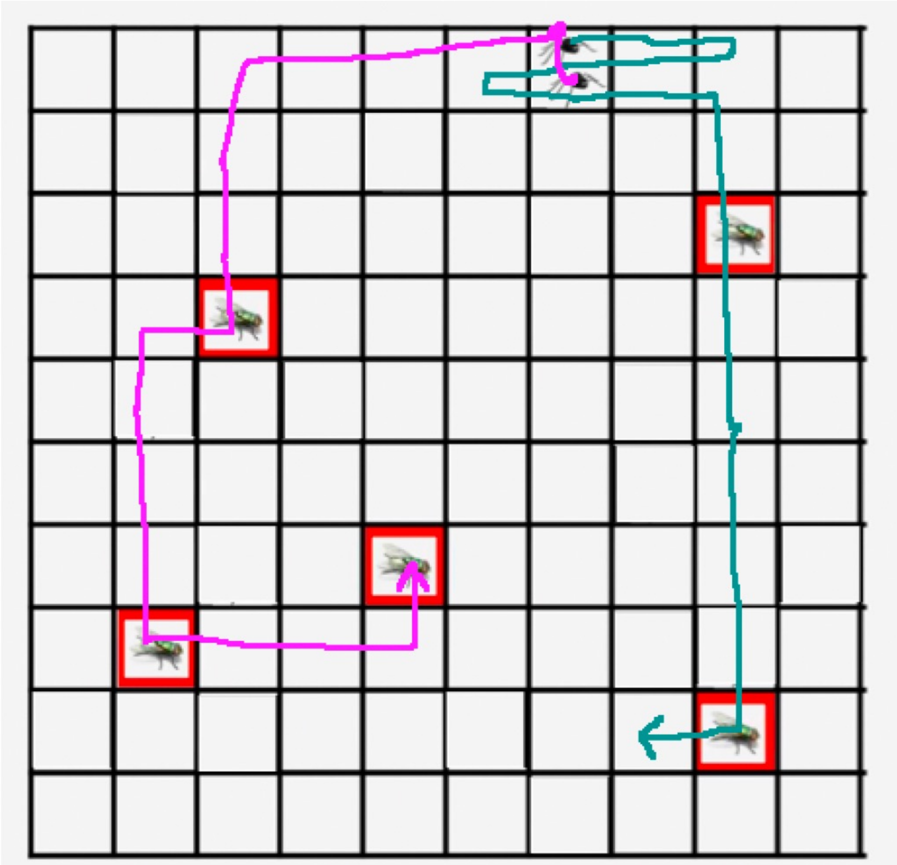
To summarize, the following is the comparison of the three methods that we tried.

	Base heuristic	**Standard rollout**	**Multiagent rollout**
Computation time (ms)	1.88	21.42	9.76
#steps in the solution	25	17	17

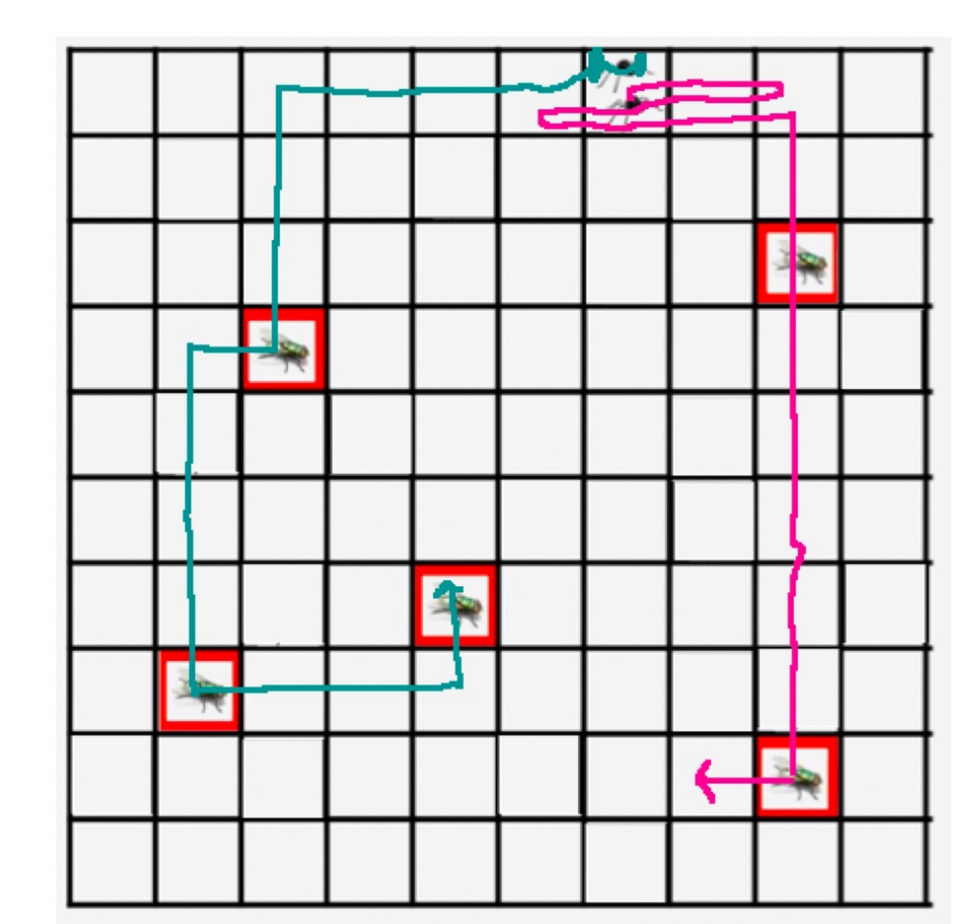
Results – standard rollout and multiagent rollout

The **computation took around 21.42 milliseconds** in this case.
this policy takes **17 steps** to take all the flies down, which is 8 ste

In the following figure, we illustrate both spiders' paths.



Time: 9.76 ms
Steps: 17



Thank You!

Email: a.mamun@asu.edu
Website: abdullah-mamun.com