

GitHub Supplement to “An MILP-Based Solution Scheme for Factored Markov Decision Processes”

This document accompanies the paper “An MILP-Based Solution Scheme for Factored Markov Decision Processes.” It provides details on the problem setting and algorithmic implementation used in our numerical experiments (Sections 1 and 2), as well as additional results for the system administrator problem (Section 3). Throughout, we use the notation and numbering conventions of the main paper.

1 Problem Setting

The system administrator problem from Section 6.1 of the main paper is defined by the following data:

- Each computer has three states: inoperative ($s_n = 0$), semi-operational ($s_n = 1$), and fully operational ($s_n = 2$).
- The initial distribution q is uniform over \mathcal{S} , that is,

$$q(s) = \prod_{n \in [N]} q_n(s_n), \quad \text{where} \quad q_n(s_n) = \frac{1}{3} \quad \forall s_n \in \{0, 1, 2\}, \quad \forall n \in [N].$$

- The reward function decomposes as

$$r(s, a) = \sum_{n=1}^N r_n(s_n), \quad \text{where} \quad r_n(s_n) = s_n \quad \forall s_n \in \{0, 1, 2\}, \quad \forall n \in [N].$$

- For each computer n , if the system administrator does not reboot it ($a_n = 0$), then its transition probabilities are given by

$$p_n(s'_n | s, a) = f(s'_n; s_n, s_j, s_k), \quad (1)$$

where s_j and s_k denote the states of the predecessors of computer n in the topology. In all network topologies shown in Figure 2 of the main paper, each computer has at most two predecessors. If a computer has only one (zero) predecessor(s), we set $s_j = s_n$ (and $s_k = s_n$). The function f is given by

$$f(s'_n; s_n, s_j, s_k) = \begin{cases} 0.95 \cdot \left(\frac{3}{2}\right)^{4s_n + s_j + s_k - 12} & \text{if } s'_n = 2, \\ 1 - 0.95 \cdot \left(\frac{3}{2}\right)^{4s_n + s_j + s_k - 12} - 0.95 \cdot \left(\frac{3}{2}\right)^{-4s_n - s_j - s_k} & \text{if } s'_n = 1, \\ 0.95 \cdot \left(\frac{3}{2}\right)^{-4s_n - s_j - s_k} & \text{if } s'_n = 0. \end{cases}$$

- The discount factor is $\gamma = 0.95$.

2 Inexact Cutting Plane Scheme

Algorithm 1 of the main paper is implemented via the inexact variant described in Algorithm 1 below. In each outer iteration, the algorithm first performs L rounds of heuristic constraint generation using coordinate-wise descent (cf. Remark 1 of the main paper), followed by a master problem solve. If the objective value has not improved sufficiently, the subproblem is solved as an MILP (cf. Theorem 4 of the main paper) subject to an early termination condition.

Algorithm 1 Inexact Cutting Plane Scheme to Solve Problem (3)

```
1: Input: Constraint set  $\mathcal{C}$ , parameters  $T, L, \epsilon, \delta, \tau_{\min}, \tau_{\max}$ .
2: for  $t = 1, 2, \dots, T$  do
3:   for  $l = 1, 2, \dots, L$  do
4:     Randomly generate a state-action pair  $(s, a) \in \mathcal{S} \times \mathcal{A}$ .
5:     Apply coordinate-wise descent to identify a state-action pair  $(\tilde{s}, \tilde{a})$  with a larger constraint violation.
6:     Add  $(\tilde{s}, \tilde{a})$  to the constraint set  $\mathcal{C}$ .
7:   end for
8:   Solve the master problem to obtain the weight vector  $w^t$  and record the objective value  $v^t$ .
9:   if  $v^t \leq (1 + \epsilon) v^{t-1}$  then
10:    Solve the subproblem as an MILP with early termination  $\text{ET}(\tau_{\min}, \tau_{\max}, \delta \cdot v^t)$ .
11:    if a constraint  $(s, a)$  with a sufficiently large violation is identified then
12:      Add  $(s, a)$  to the constraint set  $\mathcal{C}$ .
13:      Re-solve the master problem and update  $w^t$ .
14:    else
15:      Terminate.
16:    end if
17:  end if
18: end for
```

The early termination condition $\text{ET}(\tau_{\min}, \tau_{\max}, \delta \cdot v^t)$ is triggered once the solution time exceeds τ_{\min} seconds and one of the following conditions is met:

- (i) Gurobi has identified a constraint with violation exceeding $\delta \cdot v^t$;
- (ii) Gurobi has concluded that all constraint violations are less than $\delta \cdot v^t$;
- (iii) the time limit of τ_{\max} seconds has been exceeded.

Let N denote the number of computers in the system administrator problem. The parameter values used in our experiments are summarized in Table 1.

Table 1. Parameter values for Algorithm 1.

T	L	ϵ	δ	τ_{\min}	τ_{\max}
$20N$	$10N$	10^{-4}	10^{-3}	10	$100 + 3N$

3 Additional Results for the System Administrator Problem

Tables 1 and 2 of the main paper compare the policies and runtimes of our cutting plane approach with those of the variable-elimination method, an approximate dynamic programming (‘ADP’) scheme, and the primal-dual (‘P-D’) approach on the bidirectional ring and ring of rings topologies, respectively. In this section, we report analogous results for the remaining three topologies: star (Table 2), 3-leg (Table 3), and ring-and-star (Table 4).

The results are qualitatively consistent with those reported in the main paper. Across all three topologies, both the primal-dual approach and the approximate dynamic programming scheme produce substantially inferior policies compared to our cutting plane approach, and they are outperformed by the naïve benchmark strategies on larger problem instances. We note that these three topologies are relatively simple and highly structured. In the star topology, for instance, the central node naturally has the highest priority, while in the 3-leg topology, node priority decreases monotonically with distance from the central node. Due to this clear structural pattern, the ‘level’ heuristic can effectively exploit the topology and, in some cases, outperform our scope-1 value function approximation.

	12	24	36	48	60
P-D (scope-1)	294.4	574.9	839.2	1,082.1	1,322.1
P-D (scope-2)	373.0	629.5	908.7	1,195.3	1,741.5
ADP (scope-1)	404.1	712.7	1,036.4	1,094.5	1,325.4
ADP (scope-2)	409.4	742.5	962.7	1,093.5	1,392.3
ADP (scope-3)	417.9	753.1	956.0	1,084.2	1,392.7
random	421.4	752.8	1,027.9	1,279.9	1,525.2
priority	425.0	781.2	1,096.3	1,374.6	1,638.6
level	434.2	830.9	1,210.1	1,570.4	1,921.3
ours (scope-1)	437.0 (3.8%)	836.7 (5.3%)	1,205.1 (6.6%)	1,566.0 (9.2%)	1,910.9 (8.3%)
ours (scope-2)	437.3 (3.4%)	838.8 (4.6%)	1,213.4 (5.7%)	1,577.1 (7.8%)	1,927.5 (6.9%)
ours (scope-3)	437.5 (2.2%)	839.6 (3.9%)	1,215.3 (4.9%)	1,583.2 (4.6%)	1,932.3 (4.2%)

Table 2. Expected total discounted reward for the approximate dynamic programming scheme (‘ADP’), the primal-dual approach (‘P-D’), three naïve benchmark strategies (‘random’, ‘priority’ and ‘level’) as well as our cutting plane approach (‘ours’) on the star topology. In contrast to the benchmark strategies, our approach additionally provides optimality gaps, which are listed as well.

	12	24	36	48	60
P-D (scope-1)	328.4	573.1	813.3	1,052.7	1,292.9
P-D (scope-2)	332.3	619.9	862.6	1,104.2	1,346.6
ADP (scope-1)	411.5	709.2	891.6	1,142.9	1,383.0
ADP (scope-2)	413.2	718.4	898.5	1,143.6	1,386.1
ADP (scope-3)	420.5	718.3	896.8	1,142.2	1,387.8
random	422.3	737.7	959.6	1,187.2	1,421.0
priority	424.8	758.2	1,008.9	1,254.5	1,498.8
level	432.1	794.5	1,069.7	1,301.5	1,538.5
ours (scope-1)	431.0 (6.1%)	789.6 (10.1%)	1,056.5 (12.6%)	1,292.4 (13.2%)	1,530.8 (22.7%)
ours (scope-2)	432.5 (4.5%)	801.3 (6.6%)	1,073.1 (7.1%)	1,302.1 (7.9%)	1,546.6 (9.1%)
ours (scope-3)	433.2 (2.3%)	804.1 (4.3%)	1,082.3 (2.7%)	1,312.7 (3.4%)	1,556.3 (4.2%)

Table 3. Expected total discounted reward for the approximate dynamic programming scheme (‘ADP’), the primal-dual approach (‘P-D’), three naïve benchmark strategies (‘random’, ‘priority’ and ‘level’) as well as our cutting plane approach (‘ours’) on the 3-leg topology. In contrast to the benchmark strategies, our approach additionally provides optimality gaps, which are listed as well.

	12	24	36	48	60
P-D (scope-1)	341.3	581.6	821.2	1,060.3	1,299.3
P-D (scope-2)	374.4	572.8	854.8	1,073.8	1,316.7
ADP (scope-1)	410.2	573.0	814.9	1,089.1	1,298.2
ADP (scope-2)	410.8	619.1	842.7	1,101.7	1,361.8
ADP (scope-3)	415.8	617.6	838.8	1,102.8	1,376.6
random	407.2	688.3	924.9	1,166.1	1,386.2
priority	411.0	713.4	973.9	1,233.2	1,407.4
level	422.1	765.9	1,064.1	1,359.5	1,657.8
ours (scope-1)	426.4 (5.8%)	773.2 (8.9%)	1,074.9 (10.2%)	1,355.4 (11.9%)	1,650.5 (13.5%)
ours (scope-2)	426.7 (4.4%)	789.8 (6.6%)	1,086.8 (7.6%)	1,390.3 (6.7%)	1,689.4 (8.7%)
ours (scope-3)	427.0 (2.8%)	797.3 (4.3%)	1,095.3 (4.4%)	1,399.7 (3.3%)	1,708.3 (3.0%)

Table 4. Expected total discounted reward for the approximate dynamic programming scheme (‘ADP’), the primal-dual approach (‘P-D’), three naïve benchmark strategies (‘random’, ‘priority’ and ‘level’) as well as our cutting plane approach (‘ours’) on the ring-and-star topology. In contrast to the benchmark strategies, our approach additionally provides optimality gaps, which are listed as well.