the current alternate agent action, and there are \( O(N) \)
possibilities. The product of these quantities yields a
state space of \( O(N^2 K^3) \). For similar arguments as
given in Section 5, we can construct optimal policies
that are NONRECUR-AbsorB for SEQ-MDP and use
planning methods similar to those described in Section
5. There are \( O(K) \) possible actions for each state in a
SEQ-MDP. Therefore, again excluding costs for com-
puting the marginal probabilities, the computational
cost of computing an optimal infinite-horizon policy
for a SEQ-MDP is \( O(N^2 K^4) \), and the finite horizon
\( H \) cost is \( O(\min(H, NK) N^2 K^4) \).

We now prove that computing the optimal policy for a
SEQ-MDP yields a boundedly-optimal policy for the
original IDP-MDP with \( N > 1 \).

**Theorem 6.2.** An optimal policy for a SEQ-MDP
that is a transformation of an IDP-MDP with \( N > 1 \),
has an expected cost \( V^{\text{seq}} \) which is bounded by
\( V^{\text{seq}} = V^* + \sum_{k=1}^K (\delta_k - \delta_1) + N(c_{N+1} - c_1) \),
where \( V^* \) is the optimal value for the IDP-MDP.

**Proof.** We will make a constructive argument by pro-
viding a policy \( \pi \in \Pi \) that realizes the bound \( V^{\text{seq}} \).
Consider a policy \( \pi \) that starts with the lowest cost ac-
ction, and offers the highest possible incentive. The pol-
cy proceeds by sequentially decreasing the offered in-
centive \( \delta_k \) for alternate agent action \( a_k \) until the agent
rejects the offered incentive. Let this highest incentive
at which the agent rejects be \( \delta_j \). That means
that the true incentive \( t_j \) for alternate agent action \( a_j \) is
\( \delta_j \). Since \( a_j \) is the lowest cost and lowest reward
alternate action, we know that \( \delta_j \) will be accepted for
all other alternate agent actions \( i > j \), since the true
incentives exceed the offered incentive. Therefore
we need not offer \( \delta_j \) for any other agent actions. The
policy then moves on to offering incentives for alter-
mate agent action \( a_2 \), starting with offering \( \delta_{j-1} \).
The policy again continues to incrementally decrease the
offered incentive until it reaches a reject, at which it
then starts to provide incentives for \( a_3 \), and so on. The
policy is similarly defined for all subsequent alternate
agent actions. The defined policy \( \pi \) operates only on
the SEQ-MDP version of the IDP-MDP.

Note that after the principal makes an offer, either
an incentive for a particular action, or a particular
incentive for all actions can be eliminated. Since there
can at most be \( K \) accepts, and at most \( N \) rejects (as at
most one offer can be rejected for each action), there
are at most \( K + N \) steps.

Each of the at most \( N \) rejected offers will cause the
algorithm to incur at most \( c_{N+1} - c_1 \) additional cost
compared to the optimal policy, since the principal will
pay the cost of the default action \( c_{N+1} \). Each offer
that was accepted that is above the true incentive will
result in an additional cost of at most \( \delta_i - \delta_1 \). Note that
each incentive \( \delta_i \) is offered only once if it is accepted
for an action \( a_n \). Therefore, the total cost is bounded by
\( V^{\text{seq}} = V^* + N(c_{N+1} - c_1) + \sum_{k=1}^K (\delta_i - \delta_1) \).
This is true irrespective of the distribution over incentives. It can be
easily seen that the policy \( \pi \in \Pi \), as required.

This bound does not use any information about the
probability distribution. A tighter bound using the
probability distribution can be obtained but is not
presented here for ease of exposition. Also, an
alternate algorithm with bound \( V^* + K(c_{N+1} - c_1) +
\sum_{i=1}^N (c_N - c_i) \) is possible by starting from action \( a_N \)
rather than \( a_1 \) and progressing to lower cost actions
only after identifying the true incentive for the higher
cost actions. An argument similar to the one made in
Theorem 6.2 can be used to prove this bound. In next
section, our empirical results show the optimal policy
for a SEQ-MDP performs well in practice.

7 Experiments

We now empirically evaluate our algorithms.

In our simulations the cost of the default agent action
was set to \( c_{N+1} = 2 \). For \( N \) alternate agent actions,
the cost to the principal of the agent taking action \( a_n \)
was set to \( c_n = \left( \frac{N}{N} \right)^\eta \) where \( \eta \) is a constant. For \( K \)
incentives, the value of the \( k \)-th incentive was \( \delta_k = k/K \).
Note the maximum cost of the alternate agent actions
is 1, and the maximum incentive is 1. Therefore the
cost to the principal of an agent accepting an incentive
\( c_n + \delta_k \) is always less than or equal to the cost to
the principal of the agent’s default action \( c_{N+1} \), as our
IDP model assumed (Section 4). The initial belief was
set to a uniform joint distribution over the possible
incentive-alternate agent action space: recall that this
distribution must always respect the constraints that
\( t_i \geq t_j \) for \( i < j \) since \( R_{a_k} < R_{a_j} \). In each experimental
run we fixed \( K, N, \eta \) and the horizon \( H \), and sampled a
true hidden vector of incentives for the agent from the
initial belief. We then executed the policy of each al-
gorithm and recorded the total cost accumulated over
\( H \) steps. We simulated 1000 runs and averaged over
each run’s total sum of costs. We repeated this for 10
rounds (each of 1000 runs) and computed the standard
deviation error bars for the average total cost.

We compare the performance of our algorithms to two
natural methods. The first is a greedy algorithm. In
the greedy algorithm, given a current state \( B_S \) (with
\( S_n = (s_n, c_n) \)), the principal offers the incentive \( \delta_k \)
for alternate action \( a_n \) that minimizes the expected
immediate cost to the principal,

\[
[\delta_k, a_n] = \arg \min_{\delta_k, a_n} \delta_k, a_n \sum_{k'=s_n}^k p_{k', S}^n(\delta_k + c_n) + \sum_{k'=k+1} c_n p_{k', S}^n(c_{N+1}).
\]