Multiobjective Optimization in Control with Communication for Decentralized Discrete-Event Systems

W.H. Sadid, S.L. Ricker and S. Hashtrudi-Zad

Abstract—The trade-off between the cost and accuracy of a decentralized discrete-event control solution with synchronously communicating controllers is explored as a multiobjective optimization problem. We examine a class of problems where communication protocols are necessary to realize the exact control solution. In certain circumstances, it may be advantageous, from a cost perspective, to reduce communication, but incur a penalty for synthesizing an approximate control solution. A widely-used evolutionary algorithm (NSGA-II) is adapted to examine the set of Pareto-optimal solutions that arise for this family of decentralized discrete-event systems.

I. INTRODUCTION

Quantitative optimal control has been examined from the perspective of centralized discrete-event control [9], [12], [21]. Costs are assigned to control decisions, and the goal is to synthesize a controller with an overall minimal cost with respect to the control strategy. An alternate technique for measuring the cost of centralized control was introduced in [16]. Although not developed with control theory applications in mind, a new class of quantitative languages (based on weighted automata) has also been proposed [3]. Optimal decentralized control in the absence of communication, using Nash equilibrium as the optimization criterion, was studied in [13]. In [23] the notion of fictitious play is employed to find quantitatively optimal decentralized control strategies for an intruder/detection problem. An algorithm for calculating Nash equilibrium of multi-agent systems was adapted for the quantitative analysis of communication protocols in decentralized discrete-event control [20].

We are interested in a class of quantitative decentralized discrete-event control problems where we want to optimize more than one function or objective: giving rise to a multiobjective optimization problem [22]. When incorporating communication into the decentralized control problem, there may be a cost advantage to synthesizing only part of the specification, instead of realizing the entire specification with a costly communication protocol. To that end we want to investigate the trade-off between the cost of an exact control solution achieved with communication and an approximate solution, where penalties are assessed for achieving a sublanguage of a desired controllable and observable specification, with a possibly cheaper communication policy. We examine our multiobjective optimization problem using evolutionary algorithms [1].

Evolutionary algorithms, inspired by biological processes, are ideal for optimization problems when (i) exhaustive search is computationally prohibitive, and (ii) there are multiple objectives to optimize. An initial population of possible solutions are considered, and a measure of their fitness determines whether a member of the population will be involved in the formulation of the next generation of the population. Just as in natural adaptation, over a period of many generations, a population of solutions evolve that are “closer” to an optimal solution than their predecessors. We use a modified version of the Non-dominated Sorting Genetic Algorithm (NSGA-II) [5], which has already proven useful for a diverse range of control problems (e.g., [7], [24]).

The paper is organized as follows. The next section contains terminology and notation of discrete-event systems, decentralized admissible control laws and communication protocols. Decentralized control and communication costs are defined in Section III to present the multiobjective optimization problem in decentralized discrete-event systems. An example is given in this section to find the trade-off between the cost and precision of a control solution using an evolutionary algorithm.

II. BACKGROUND

Supervisory control of discrete-event systems proposed in [15] uses formal language theory to model the behavior of an uncontrolled system as well as the desired behavior (specification) for the controlled system. Specifically, the system behavior is described by a regular language \(L\), which can be represented by a finite automaton, \(M_L\):

\[
M_L = (Q, \Sigma, \delta, q_0, Q_m),
\]

where \(Q\) is a finite set of states; \(\Sigma\) is a finite set of symbols called the alphabet; \(\delta\) is the transition function defined as \(\delta : Q \times \Sigma \rightarrow Q\) and we write \(\delta(q, \sigma)\) when \(q' \in Q\) such that \(\delta(q, \sigma) = q'\); \(q_0\) is the initial state; and \(Q_m \subseteq Q\) is a set of marked states. The specification is denoted by the language \(K\), where \(K \subseteq L \subseteq \Sigma^*\), and an automaton marking \(K\) is \(M_K\). Finite automata are also used to model the controllers that issue commands to ensure that the system adheres to a given specification. We are interested in systems where \(n\) controllers (let \(I = \{1, \ldots, n\}\)) independently issue control commands to ensure that the specification is met.

The prefix closure of a language \(K\) is defined as \(\overline{K} := \{s \in \Sigma^* \mid (\exists t \in \Sigma^*) \text{ such that } st \in K\}\). When \(K\) is prefix-closed \(K = \overline{K}\). The marked language of \(M_L\), denoted by
$L_m$, is defined as $L_m := \{ s \in L \mid \delta(q_0, s) = q' \land q' \in Q_m \}$. $K$ is said to be $L_m$-closed if $K = K \cap L_m$.

The synchronous product of two automata $M_i = (Q_i, \Sigma, \delta_i, q_{0i}, Q_{mi})$, for $i = \{1, 2\}$, is denoted by $M_1 || M_2$ and is defined as the reachable subgenerator of the automaton $M_1 \times M_2 = (Q_1 \times Q_2, \Sigma_1 \cup \Sigma_2, \delta_{1||2}, (q_{01}, q_{02}), Q_{m1} \times Q_{m2})$, where

$$
\delta_{1||2}((q_1, q_2), \sigma) =
\begin{cases}
    (\delta_1(q_1, \sigma), \delta_2(q_2, \sigma)), & \text{if } \delta_1(q_1, \sigma) \land \delta_2(q_2, \sigma)\!; \\
    \emptyset, & \text{otherwise}.
\end{cases}
$$

In the context of the supervisory control problem, $\Sigma$ is partitioned into two sets for each controller: controllable events that can be prevent from occurring, denoted by $\Sigma_{uc,i}$ (for $i \in I$) and uncontrollable events that controller $i$ cannot prevent from occurring, denoted by $\Sigma_{uo,i}$. The overall set of controllable events is $\Sigma_c := \bigcup_{i=1}^{n} \Sigma_{c,i}$ and the overall set of uncontrollable events is $\Sigma_u := \bigcup_{i=1}^{n} \Sigma_{u,i}$. For all $\sigma \in \Sigma_c$, we define $I_c(\sigma) = \{ i \in I \mid \sigma \in \Sigma_{uc,i} \}$. The specification $K$ is controllable wrt $L$ and $\Sigma_{uc}$ if

$$
K \subseteq \Sigma_{uc} \cap L \subseteq K.
$$

Another aspect of the control problem involves the notion of partial observation: there are some events that controller $i$ can observe, namely $\Sigma_{o,i}$ for $i \in I$, while the rest of the events in $\Sigma$ are unobservable to controller $i$, denoted by $\Sigma_{uo,i}$.

To formally capture the notion of partial observation, we define a canonical projection $\pi_i : \Sigma^* \to \Sigma_{o,i}^*$. Thus for $t = \sigma_1 \sigma_2 \ldots \sigma_m \in \Sigma^*$, the partial observation $\pi_i(t)$ will contain only those events $\sigma_i \in \Sigma_{o,i}$, since unobservable events are removed. The specification $K$ is co-observable [19] wrt $L$, $\Sigma_{o,i}$, and $\Sigma_{c,i}$ if

$$
(\forall t \in K)(\forall \sigma \in \Sigma_c) \exists \sigma' \in L \setminus \overline{K} \Rightarrow \exists i \in I_c(\sigma) \pi_i^{-1}(\sigma) \cap \overline{K} = 0.
$$

When $I = \{1\}$, $K$ is said to be observable [10].

A decentralized control law for controller $i$ is a mapping $\Gamma_i : \pi_i(L) \to \text{Pwr}(\Sigma)$ that defines the set of events that controller $i$ believes should be enabled based on its partial view of the system behavior. While controller $i$ can choose to enable or disable events in $\Sigma_{c,i}$, it must enable all events in $\Sigma_{uc,i}$.

$$
(\forall i \in I)(\forall t' \in L) \Gamma_i(\pi_i(t')) = \{ \gamma \in \text{Pwr}(\Sigma) \mid \gamma \supseteq \Sigma_{uc,i} \}.
$$

Admissible decentralized control laws $\Gamma_i$ allow local decisions to be taken in an observationally-equivalent fashion:

$$
(\forall t, t' \in L)(\forall i \in I) \pi_i(t) = \pi_i(t') \Rightarrow (\sigma \in \Gamma_i(t) \Rightarrow \sigma \in \Gamma_i(t')).
$$

To find a solution to the decentralized control problem in the absence of communication between controllers, we want to find $\Gamma_i (\forall i \in I)$ such that $\forall i \in K$:

$$
(\forall \sigma \in (\Sigma_c \cup \Sigma_{uc})) \forall \sigma \in K \Rightarrow \sigma \in \bigcap_{i \in I} \Gamma_i(\pi_i(t)) \\
(\forall \sigma \in \Sigma_c) \forall \sigma \in L \setminus \overline{K} \Rightarrow \sigma \notin \bigcap_{i \in I} \Gamma_i(\pi_i(t)).
$$

From the results of [19], such $\Gamma_i (\forall i \in I)$ exist if the specification $K$ is co-observable (wrt $L$, $\Sigma_{o,i}$ and $\Sigma_{c,i}$), controllable (wrt $L$ and $\Sigma_{uc}$), and $L_m$-closed.

When $K$ does not satisfy Eq. (2) and $n \geq 2$, it may still be possible to find a control solution by introducing synchronous communication between controllers. We know from [2], [18] that we can synthesize synchronous communication protocols when $K$ is controllable (wrt $L$ and $\Sigma_{uc}$), $L_m$-closed, observable (wrt $L$, $\Sigma_{o}$ and $\Sigma_{c}$) but is not co-observable (wrt $L$, $\Sigma_{o,i}$ and $\Sigma_{c,i}$). From now on in this paper, we assume that $K$ is controllable, observable and $L_m$-closed.

The synthesis of communication protocols requires the introduction of a set of messages $\Delta$ that controllers send to each other. Let $\Delta = \bigcup_{i,j} \Delta_{i,j}$, where $a \in \Delta_{i,j}$ is a message that controller $i$ sends to controller $j$. For the problem that we consider here, $\Delta_{i,j} \subseteq \Sigma_{o,i} \setminus \Sigma_{o,j}$. It could be the case that no message is sent, in which case the controller is silent ($\epsilon$). Let $\Delta_{i,j}^t := \Delta_{i,j} \cup \{ \epsilon \}$. A synchronous communication protocol between controllers $i, j \in I$ is a mapping $\phi_{i,j} : \Gamma_i \to \Delta_{i,j}^t$ and indicates the message that is synchronously sent from controller $i$ to controller $j$.

The latest information that a controller can keep through a sequence can be defined as $\psi_i : L \to \Delta_{i,j}^t \cup (\cup_{i \neq j} \Delta_{i,j})$, such that when $t = \sigma_1 \ldots \sigma_m \in L$ occurs, each controller $i$ keeps track of communication it receives about $t$ along with its own observations of $t$.

$$
\psi_i(t) = \begin{cases}
    \sigma_m, & \text{if } \sigma_m \in \Sigma_{o,i} \text{ or } \\
    \sigma_m \notin \Sigma_{o,i} \text{ and } \exists j \in I \text{ s.t. } \phi_{j,i}(t) \neq \epsilon; \\
    \epsilon, & \text{otherwise}.
\end{cases}
$$

The canonical projection $\pi$ is extended to include received messages: $\pi_i^\Delta : \Sigma^* \to (\Sigma_{o,i} \cup (\cup_{i \neq j} \Delta_{i,j}))^*$, where $\pi_i^\Delta(\epsilon) = \epsilon$, and $\pi_i^\Delta(t) = \psi_i(\sigma_1) \psi_i(\sigma_1 \sigma_2) \ldots \psi_i(\sigma_1 \ldots \sigma_m)$, for $t = \sigma_1 \ldots \sigma_m$.

Finally, it must be the case that communication occurs in an observationally-equivalent manner. Communication protocols $\phi_{i,j}$ are admissible if

$$
(\forall t, t' \in L)(\forall i \in I) \pi_i(t) = \pi_i^\Delta(t) \Rightarrow \phi_{i,j}(t) = \phi_{i,j}(t')
$$

We extend the decentralized control law to a communicating controller $i$ as follows $\Gamma_i^\Delta : \pi_i^\Delta(L) \to \text{Pwr}(\Sigma)$. To find a solution to the decentralized control problem with synchronous communication protocols $\Phi = \{ \phi_{i,j} \}$ for all controllers $i, j \in I$, we have to find $\Gamma_i^\Delta (\forall i \in I)$ such that
∀t ∈ K:
(∀σ ∈ (Σc ∨ Σuc)) tσ ∈ K ⇒ σ ∈ ∩i∈I Γ∆ni (∆ni (t)) ∧
(∀σ ∈ Σc) tσ ∈ L \ K ⇒ σ ̸∈ ∩i∈I Γ∆ni (∆ni (t)). \quad (5)

From the results of [17], we can find such Γi when K is
coco-observable with respect to L, πi, and Σc,i, controllable
(wrt L and Σuc,i), and Lm-closed.

III. MULTIOBJECTIVE OPTIMIZATION OF
DECENTRALIZED DES WITH COMMUNICATION

A multiobjective optimization problem is characterized by
the requirement to optimize multiple conflicting objectives.
Evolutionary algorithms are used for solving multiobjective
optimization problems. The idea of such algorithms is as
follows: beginning with an initial population of possible
solutions, each solution is assigned a fitness value indicating
its quality. The fitness value determines which solutions will
be selected for breeding the next generation. These candi-
dates are mutated and combined to produce new “children”
candidate solutions. The evolutionary process continues until
either an optimal set of solutions is determined or a pre-
determined number of generations is exceeded.

There may not exist a single best solution in the multi-
objective optimization problem. Instead, evolutionary algo-
rithms define a set of best solutions. The class of evolutionary
algorithms that we are using produces a Pareto front of
the candidate solutions. Solutions that comprise the front
are said to be Pareto-optimal or non-dominated. A solution
x1 is said to be dominated by another solution x2, if x1 is
not better than x2 in any objectives, and x1 is strictly
worse than x2 in at least one objective. Most evolution-
ary multiobjective optimization approaches such as strength
Pareto evolutionary algorithm (SPEA) [25], non-dominated
sorting genetic algorithms (NSGA-II) [5], and the Pareto-
archived evolution strategy (PAES) [8] use the concept of
dominance. To solve our multiobjective DES problem, we
use NSGA-II. Unlike some of the other approaches, NSGA-
II keeps an archive of the best b solutions generated so far:
all children of generation k compete for membership in
generation k + 1 with generation k. In this way, good
solutions from a previous generation are preserved. The
algorithm also features a strong fitness assignment procedure.

For decentralized DES with communication, we have two
objectives to optimize: each decentralized controller i must
optimize the cost of its local control law vi and the cost of
its local communication policy ui. Ideally, we would
like the joint decisions of the controllers in the presence
of the full communication protocol to allow exactly K to
occur; however, in the presence of a costly communication
protocol, it might be more efficient to allow some subset
of K to occur. But it may be the case that the penalty
for disabling certain sequences within K is more than the
communication required to enable the same sequence. We are
interested in a quantitative analysis of the trade-off between
the cost of imperfectly controlling the system by removing
some (potentially costly) communications and the cost of
taking exact control solution with the full communication
protocol.

We adapt the centralized control cost function of [21]
to the case of the control cost function for a decentralized
controller i (for i ∈ I). We consider three basic costs that
controller i (for i ∈ I) can incur to control a system:

1) We assume that there is a basic cost for an event to occur, which can be considered to be the cost to enable
an event, denoted by ei : Σ → R+ ∪ {0}.

2) There is a cost to disable an event that would otherwise
take the system out of K, di : Σ → R+ ∪ {0, ∞}.

3) Since our control objective is to have the collection
of Γi (for i ∈ I) allow exactly K to occur, when a
transition is disabled that would otherwise keep the
system in K, the cost to disable is incurred, plus an
additional penalty is assessed: pK ∈ R.

We assume that a disablement (and any associated penalty) or
an enablement cost lies in the range of [0, ∞). When a con-
troller tries to disable an uncontrollable event, a penalty of
∞ is levied. When a controller is not sure whether or not the
system leaves K via arollable event, corresponding to an
“uncertain” decision for controller i, the default decision is to
enable the event. We consider the cost of an uncertain control
decision to be cost of enablement. Because we will consider
only control laws that keep the system within K, it will not
be possible that all controllers enable same event that takes
the system out of K. Note that the costs considered here are
associated with a controller’s local decision regarding the
occurrence of an event, and not for the eventual fusion of the
control decisions.

The control cost v1 : Γi × K × Σ → R+ ∪ {0, ∞} describes
the cost incurred by controller i for the occurrence of event
σ ∈ Σ for t ∈ K such that δ(g0, tσ)!

\[ v_i(Γ_i, t, σ) = \begin{cases} 
    e_i(σ), & \text{if } σ ∈ Γ_i π_i(π_i(t)); \\
    d_i(σ), & \text{if } σ ̸∈ Γ_i π_i(π_i(t)) \text{ and } π_i = π_i(π_i(t))\cap K = \emptyset; \\
    d_i(σ) + p_K, & \text{if } σ ̸∈ Γ_i π_i(π_i(t)) \text{ and } π_i = π_i(π_i(t))\cap K \subseteq K; \\
    ∞, & \text{otherwise.} 
\end{cases} \quad (6) \]

The total control cost for controller i (for i ∈ I) is then
\[ V_i(Γ_i, K, Σ) = \sum_{t ∈ K} \sum_{σ ∈ Σ} v_i(Γ_i, t, σ). \]

Each decentralized controller i has a communication proto-
ocol Φi = {Φi,1, ..., Φi,n}. We assume that a basic
cost for communication is incurred each time controller i
sends a message to controller j, denoted by comi : Σ → R+ ∪ {0}. The cost of controller i’s communication protocol
ui : Φi × K × Σ → R+ ∪ {0} assumes that a cost is incurred
only when a communication is sent by controller i:

\[ u_i(Φ_i, t, σ) = \begin{cases} \text{comi}(σ), & \text{if } (∃j ∈ I) Φ_i, j(π_i(t)) = σ; \\
0, & \text{otherwise.} \end{cases} \quad (7) \]

It is possible that two identical messages sent by different
controllers incur different local costs. There is no cost for
the reception of a message and it may be the case that
the cost for a point-to-point communication differs from
that of a broadcast. For simplicity, we assume that when controller \( i \) communicates the same message to more than one controller, a single cost is incurred, regardless of the number of recipients.

The communication cost for all \( t \in \mathcal{K} \) for controller \( i \) (for \( i \in I \)) is then \( U_i(\Phi_i, \mathcal{K}, \Sigma) = \sum_{t \in \mathcal{K}, \sigma \in \Sigma} u_{ij}(\Phi_i, t, \sigma) \). Note that when considering the costs for control and communication in cyclic systems, the total cost function should be updated accordingly to those calculating average cost [17] over an infinite horizon.

To ensure that the objective functions are defined across the same domain, we adjust the definition of \( U_i \) and \( V_i \) accordingly, so that both functions are defined over \( \Gamma^\Delta_i \times \Phi_i \times \mathcal{K} \times \Sigma \).

Objective 1: The first objective function is the cost of the control decisions each controller \( i \in I \) makes for its observation of \( \mathcal{K} \):

\[
o_{1,i}(\Gamma^\Delta_i, \Phi_i, \mathcal{K}, \Sigma) = V_i(\Gamma^\Delta_i, \Phi_i, \mathcal{K}, \Sigma).
\]

Objective 2: The second objective function is the cost of the communication protocol that each controller uses to assist the other controllers in reaching the control objective for \( \mathcal{K} \):

\[
o_{2,i}(\Gamma^\Delta_i, \Phi_i, \mathcal{K}, \Sigma) = U_i(\Gamma^\Delta_i, \Phi_i, \mathcal{K}, \Sigma).
\]

We consider an optimization problem with a finite set of control laws \( \Gamma^\Delta \), and a finite set of communication protocols \( \Phi \). We want to find the trade-off in minimizing the cost of imposing a costly communication protocol \( \Phi \) on the uncontrolled system \( L \) to reach our control objective as compared to eliminating some of the communication and taking a penalty for not reaching the control objective.

Problem 1: Given \( K \subseteq L \), find \( \Gamma^\Delta_i \) and \( \Phi_i \) \((\forall i \in I)\) to

\[
\min_{\Gamma^\Delta_i \times \Phi_i} f_i(\Gamma^\Delta_i, \Phi_i, \mathcal{K}, \Sigma) = [o_{1,i}(\Gamma^\Delta_i, \Phi_i, \mathcal{K}, \Sigma),
\]

subject to \( \emptyset \subseteq \cap_{i=1}^{n} \Gamma^\Delta_i(\pi_{\Delta}(\mathcal{K})) \subseteq K \), and \( \Gamma^\Delta = \{\Gamma^\Delta_1, \ldots, \Gamma^\Delta_n\} \) are admissible.

We address Problem 1 by applying the evolutionary algorithm NSGA-II [5]. The main algorithms required to describe NSGA-II are presented in [11]. We create an initial population of pairs of possible control laws and communication protocols \( \langle \Gamma^\Delta_i, \Phi_i \rangle \) that satisfy the constraints of Problem 1. In accordance with NSGA-II, each member of the population is assigned a fitness value, calculated wrt the values of the two objective functions. From the initial population, candidate members for the Pareto front are calculated: those members of the population that are non-dominated. The next generation is calculated following a “breeding” process of elements from the preceding generation. Admissibility of potential control and communication solutions is determined during breeding. Those members of the previous and current population with the best fitness values are then ranked and reorganized into a new candidate set for the Pareto front. This process continues until either we exceed the number of pre-specified generations or the ideal Pareto front is found.

Note that at the conclusion of the algorithm, we have a set of optimal solutions from which to choose. In particular, solutions to Problem 1 provide Pareto optimal costs with respect to communicating controller \( i \). Thus, the designer is free to choose a solution that favours one controller over another, based on the Pareto-fronts produced for each controller. One possible strategy to arrive at a set of global solutions for the locally optimal possibilities from the DES adaptation of NSGA-II is the Hierarchy Algorithm from [6].

A. Example:

Let us consider a problem in the space science where a number of robots navigate to explore an area of a planet. The area map is divided into square boxes, where the robots can move from one box to another, either horizontally (left-right), or vertically (up-down). Each movement is represented by an event, and each event occurs at a cost. The event cost in one direction may be higher than the other direction, e.g., if the surface is steep in one direction, then the robots need more energy to move than the other direction. In general, we can divide the area into \( m \times m \) square boxes. Suppose there are \( n \) robots to explore the area, and more than \( n \) target states where the robots want to reach. Their actions are subject to a single constraint: no two robots can occupy the same target state at any time.

For simplicity, in this example, we consider a \( 3 \times 3 \map (m = 3) \) and \( n = 2 \) robots, denoted by \( R_1 \) and \( R_2 \), each having 2 target states to reach. The automaton for each robot is shown in Fig. 1. An event \( xyi \in \Sigma_i \) corresponds to a move from state \( x \) to state \( y \) by \( R_i \). All events are locally controllable (e.g., \( R_i \) controls only events that end in \( i \)). Similarly, all events are locally observable (e.g., \( R_i \) observes only events that end in \( i \)). \( R_1 \) starts from state 1 and has target states 7 and 8 whereas \( R_2 \) starts from state 3 and has target states 8 and 9. According to the constraint noted above, \( R_1 \) and \( R_2 \) cannot be in state 8 at the same time.

The system behavior \( L \) is the language generated by the synchronous product of \( R_1 || R_2 \). The corresponding automaton \( M_L \) has 81 states and 234 transitions. The specification automaton \( M_K \) is a subautomaton of \( M_L \), missing only \( (8,8) \) from \( M_L \) and the transitions associated with that state.

The robots have a map of the area, but no robot knows in which target state it will end up. For instance, if \( R_1 \) reaches state 7, then \( R_2 \) can go to either state 8 or state 9. But if \( R_1 \) has already reached state 8, then \( R_2 \) must be informed about the position of \( R_1 \), so that \( R_2 \) can move to state 9. In fact, to avoid the situation when both \( R_1 \) and \( R_2 \) are in state 8, it is necessary that each robot inform the other whenever the following events occur: \( 58i, 78i \) or \( 98i \), for \( i \in \{1, 2\} \).

\footnote{We used the software DESUMA to calculate the synchronous product.}
Three basic costs are assigned for the control cost function:

\[
e_1(\sigma) = \begin{cases} 
50, & \text{if } \sigma \in \{121,141,231,251,361\}; \\
100, & \text{if } \sigma \in \{451,471,541,561,651,691\}; \\
150, & \text{if } \sigma \in \{781,981\}.
\end{cases}
\]

\[
e_2(\sigma) = \begin{cases} 
100, & \text{if } \sigma \in \{142,212,252,322,362\}; \\
150, & \text{if } \sigma \in \{452,472,542,562,582,692\}; \\
200, & \text{if } \sigma \in \{782,982\}; \\
2000, & \text{if } \sigma \in \{121,141,231,251,361\}; \\
1500, & \text{if } \sigma \in \{451,471,541,561,651,691\}; \\
1000, & \text{if } \sigma \in \{781,981\}.
\end{cases}
\]

\[
d_1(\sigma) = \begin{cases} 
500, & \text{if } \sigma \in \{581\}; \\
10000, & \text{if } \sigma \in \{781\}; \\
900, & \text{if } \sigma \in \{981\}; \\
0, & \text{otherwise}.
\end{cases}
\]

\[
d_2(\sigma) = \begin{cases} 
100, & \text{if } \sigma \in \{581\}; \\
20000, & \text{if } \sigma \in \{781\}; \\
5000, & \text{if } \sigma \in \{981\}; \\
0, & \text{otherwise}.
\end{cases}
\]

The cost of a communication is defined as below.

\[
\text{com}_1(\sigma) = \begin{cases} 
500, & \text{if } \sigma \in \{581\}; \\
10000, & \text{if } \sigma \in \{781\}; \\
900, & \text{if } \sigma \in \{981\}; \\
0, & \text{otherwise}.
\end{cases}
\]

\[
\text{com}_2(\sigma) = \begin{cases} 
500, & \text{if } \sigma \in \{582\}; \\
700, & \text{if } \sigma \in \{782\}; \\
20000, & \text{if } \sigma \in \{982\}; \\
0, & \text{otherwise}.
\end{cases}
\]

We illustrate Algorithm NSGA-II for $R_3||R_2$. The initial size of the population, $|P|$ is 40 and the algorithm was run for 100 generations. The first three ranks of the Pareto front for Controller 1 are shown in Fig. 2. The non-dominated solutions for Controller 1 (the solutions in the front of rank 1) are shown in Table I, which represents the best compromises for Robot 1. In particular, the five Pareto-optimal solutions offer a variety of possible costs for a communication policy for Controller 1, ranging from a cost of 0 up to a cost of 12,800. It is interesting to note that when Controller 1 does not communicate anything to Controller 2, the resulting solution for Controller 2 incurs a communication cost of 106,000. Whereas when Controller 1 increases its communication so that the cost of communicating $\Phi_1$ is 1000, the communication cost for Controller 2 decreases to 87,200, but the control cost increases nearly eight-fold.

The first three ranks of the Pareto front for Controller 2 are shown in Fig. 3. The non-dominated solutions for Controller 2 (the solutions in the front of rank 1) are shown in Table II, which are the best compromises for Robot 2. Again, it is interesting to examine the Pareto-optimal solutions for Controller 2: when Controller 2 does not communicate at all, the control cost for Controller 1 is 1,265,700. Whereas when Controller 2 communicates a bit more with a cost of 500,
TABLE II: Non-dominated solutions of Controller 2.

<table>
<thead>
<tr>
<th>u1(·)</th>
<th>v1(·)</th>
<th>u∗1(·)</th>
<th>v∗1(·)</th>
</tr>
</thead>
<tbody>
<tr>
<td>55,100</td>
<td>42,050</td>
<td>0</td>
<td>55,100</td>
</tr>
<tr>
<td>46,000</td>
<td>35,400</td>
<td>1900</td>
<td>36,000</td>
</tr>
<tr>
<td>26,100</td>
<td>24,000</td>
<td>2400</td>
<td>24,000</td>
</tr>
<tr>
<td>54,300</td>
<td>500</td>
<td>25,250</td>
<td>24,000</td>
</tr>
<tr>
<td>56,500</td>
<td>36,000</td>
<td>308,500</td>
<td>308,500</td>
</tr>
</tbody>
</table>

Fig. 3: Pareto fronts of rank 1,2,3 for Controller 2 after 100 generations.

the control cost for Controller 1 only goes down by about 1.3%.

Ultimately, we use NSGA-II as an initial guide to aid in the selection of local Pareto-optimal communication and control policies for decentralized DES. For instance, there may be compelling physical arguments to insist that one decentralized site assumes the bulk of the communication during the operation of system tasks, despite the site incurring a high communication cost (wrt other sites). Similarly, we may be willing for some degree of approximation on one or more sites to reduce the cost of communication to achieve a precise control decision. Modeling the trade-off as a multi-optimization problem gives us a better selection of optimal solutions from which to choose communication and control policies for this class of DES problem.

ACKNOWLEDGMENTS

The authors would like to thank Mahmudul Hasan for implementing the algorithm in Visual C++.

REFERENCES