Load Scheduling of Simple Temporal Networks Under Dynamic Resource Pricing

T. K. Satish Kumar
Information Sciences Institute
University of Southern California
tkskwork@gmail.com

Zhi Wang
Department of Computer Science
University of Southern California
zhiwang@usc.edu

Anoop Kumar **Craig Milo Rogers** **Craig A. Knoblock**
Information Sciences Institute
University of Southern California
{anoopk, rogers, knoblock}@isi.edu

Abstract

We study load scheduling of simple temporal networks (STNs) under dynamic pricing of resources. We are given a set of processes and a set of simple temporal constraints between their execution times, i.e., an STN. Each process uses a certain amount of resource for execution. The unit price of the resource is a function of time, $f(t)$. The goal is to find a schedule of a given STN that trades off makespan minimization against cost minimization within a user-specified suboptimality bound. We provide a polynomial-time algorithm for solving the load scheduling problem when $f(t)$ is piecewise constant. This has important applications in many real-world domains including the smart home and smart grid domains. We then study the dependency of the unit price of the resource on time as well as the total demand at that time. This leads to a further characterization of tractable, NP-hard, and conjectured tractable cases.

Introduction and Problem Formulation

Efficient algorithms for temporal reasoning are critical for a large number of real-world applications, including autonomous space exploration (Knight et al. 2001), domestic activity management, and job scheduling on servers (Ji, He, and Cheng 2007). Many formalisms have been proposed and are currently used for reasoning with metric time and resources (Smith and Cheng 1993; Kumar 2003; Muscettola 2004). Simple Temporal Networks (STNs) (Dechter, Meiri, and Pearl 1991) are popularly used for efficiently reasoning about *difference constraints* in scheduling problems.

Formally, An STN S is defined by a directed graph $\langle \mathcal{X}, \mathcal{E} \rangle$, where $\mathcal{X} = \{X_0, X_1, \ldots, X_N\}$ is the set of nodes representing events and $\mathcal{E}$ is the set of directed edges between them representing *simple temporal constraints*. A *schedule* τ for S is a function that maps each node to a time instant at which the corresponding event should be executed. For any schedule τ, $\tau(X_0)$ is set to 0 by convention to establish a frame of reference. Each directed edge $e_{ij} = \langle X_i, X_j \rangle \in \mathcal{E}$ is annotated with a pair of real numbers $[LB(e_{ij}), UB(e_{ij})]$, representing the simple temporal constraint $LB(e_{ij}) \leq \tau(X_j) - \tau(X_i) \leq UB(e_{ij})$. A schedule is said to be *consistent* if and only if it satisfies all constraints given by the edges in $\mathcal{E}$. Let $\Gamma(S)$ be the set of all consistent schedules of S.

Although their expressiveness is limited compared to other formalisms, STNs are widely used as they can be solved in polynomial time using shortest path computations on their *distance graph* representations. In the distance graph representation, the constraint $X_j - X_i \leq \rho$ is represented as an edge from X_i to X_j annotated with ρ. The absence of negative cost cycles in the distance graph characterizes the consistency of the temporal constraints in the STN (Dechter, Meiri, and Pearl 1991). Since the distance graph can have negative cost edges, shortest paths in the distance graph are calculated using the common Bellman-Ford algorithm. Improved algorithms for solving STNs have been developed by several authors (Xu and Choueiry 2003; Planken, de Weerdt, and van der Krogt 2008).

In this paper, we use the STN framework to study important classes of load scheduling problems that involve metric temporal constraints as well as costs of resources. Problems that can be studied in this framework include those that arise in the smart home (Qayyum et al. 2015) and smart grid domains (Sianaki, Hussain, and Tabesh 2010) as well as in high performance computing (HPC) (Yang et al. 2013) and job shop scheduling (Xiong, Sadeh, and Sycara 1992). Although the STN framework can be extended to reason about the resource requirements of *events* (Kumar 2003), in this paper, for simplicity of exposition, we reason about the resource requirements of non-preemptible *processes* represented by a pair of events that determine their starting and ending points.

Let $\mathcal{P}_S = \{P_1, P_2, \ldots, P_K\}$ be a set of processes in the STN S. We represent each $P_i \in P_S$ as a pair of events, namely $X^s_{P_i} \in \mathcal{X}$ for its starting time point and $X^e_{P_i} \in \mathcal{X}$ for its ending time point. An edge $\tilde{e}_{P_i} = \langle X^s_{P_i}, X^e_{P_i} \rangle \in \mathcal{E}$ is annotated with the bounds $[LB(\tilde{e}_{P_i}), UB(\tilde{e}_{P_i})]$ where $UB(\tilde{e}_{P_i}) = LB(\tilde{e}_{P_i}) \geq 0$. This edge represents the fixed duration of the process P_i.[1] Of course, $\mathcal{E}$ can also contain simple temporal constraints between the starting and end-

[1] It is easy to extend our framework to processes having flexible durations; see (Nattaf, Artigues, and Lopez 2017) for richer models. However, for the combinatorial problem studied in this paper, the lower bound on the duration of a process is also provably its optimal value, because longer durations require more resources. Therefore, for simplicity of exposition, we do not consider flexible durations explicitly.

ing time points of different processes. Each process P_i uses resources for execution. While these resources could be of different types, we focus on a single resource that resembles electrical energy, henceforth simply referred to as "energy". The energy consumption model could be of two different kinds: (model A) the process P_i consumes energy at a rate of w_i watts during its execution, or (model B) the process P_i demands its entire energy requirement W_i at the beginning of its execution, where $W_i = w_i \cdot [\tau(X^e_{P_i}) - \tau(X^s_{P_i})]$.

In this paper, we focus on the latter case, i.e., model B, for two reasons. First, model B is relevant for modern application domains. For example, in smart home and smart grid domains that require efficient use of renewable energy such as solar energy, demanding the entire energy requirement of a process at the beginning of its execution is beneficial for both consumers and energy providers. Given that renewable energy may not be consistently available, this mechanism benefits consumers since sufficient energy is guaranteed to be available to them for an entire process to complete. This mechanism also helps energy providers develop better demand responses in order to: (a) store energy more economically, and (b) avoid overload and power failures (U.S. Department of Energy 2006). Second, model B is better suited for a simpler explanation of the algorithmic techniques developed in this paper. As discussed later, we demonstrate that the same techniques can also be extended to other energy consumption models, including model A.

In a dynamic resource pricing model, unit prices are expected to change frequently to reflect changes in the balance between supply and demand (Borenstein 2005). Let $f(t)$ be the unit price of energy at time t. Given a consistent schedule $\tau \in \Gamma(S)$, its *cost* is given by the expression $C(\tau) = \sum_{i=1}^{K} f(\tau(X^s_{P_i})) \cdot W_i$. An optimal schedule is a consistent schedule $\tau^* \in \Gamma(S)$ such that $\forall \tau \in \Gamma(S) : C(\tau^*) \leq C(\tau)$. The goal in the load scheduling problem is to find a consistent schedule of a given STN that trades off makespan minimization against cost minimization within a user-specified suboptimality bound.

While $f(t)$ could be any function in general, it is piecewise constant in many real-world domains, including water and power supply and market prices of various goods. Figure 1, borrowed from the Southern California Edison's summer rate plan TOU-D-A as of August 2017 (Southern California Edison 2017), shows the plot of a real-world dynamic pricing of electrical energy. In this paper, we provide a polynomial-time algorithm for solving the load scheduling problem when $f(t)$ is piecewise constant. This has important applications in many real-world domains including the smart home and smart grid domains. For example, our polynomial-time algorithm is applicable to optimizing appliance scheduling in smart homes (Qayyum et al. 2015) and optimizing operations in smart home area networks to save power on a larger scale (Zhao et al. 2013). We also study the dependency of the unit price of energy on time as well as the total demand at that time. This dependency on the total demand could either encourage or penalize "buying in bulk" based on the balance of supply and demand. An analysis of this dependency leads to a further characterization of tractable, NP-hard, and conjectured tractable cases.

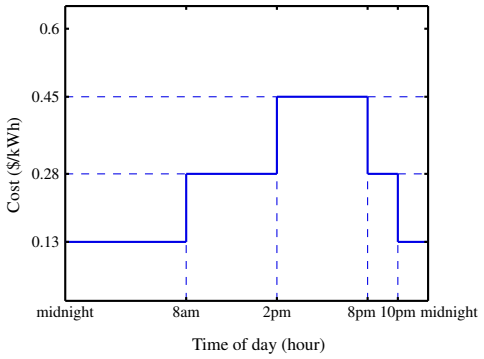


Figure 1: Shows a function $f(t)$ that models a real-world dynamic pricing of electricity (electrical energy).

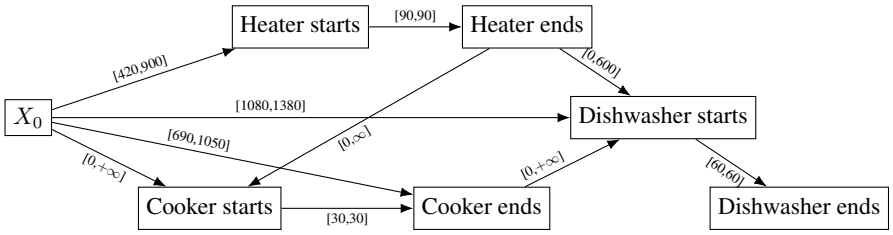
Tractable Case: Piecewise Constant $f(t)$

In this section, we present two polynomial-time procedures: Algorithm 1 and Algorithm 2. Algorithm 1 finds the optimal cost schedule for the load scheduling problem on STNs under a piecewise constant $f(t)$. Algorithm 2 uses Algorithm 1 in a quasi binary search procedure to trade off makespan minimization against cost minimization within a user-specified suboptimality bound γ. Algorithm 1 is based on a reduction to the maxflow problem on bipartite graphs and is therefore very efficient (Goldberg and Tarjan 1988). We note that the methods presented in (Morris et al. 2004) are not applicable here since $f(t)$ is not necessarily convex. We also note that the algorithm presented in (Kumar 2004) is not directly applicable here either, since the problem here is one of minimization instead of maximization. Moreover, the issue of makespan is not considered in that paper. Nonetheless, some of the basic combinatorial arguments presented in (Kumar 2004) are adapted here.

We choose a running example from appliance scheduling in smart homes to demonstrate the working of our algorithms. The running example is explained in Figure 2. Although this running example is not completely realistic if we use model B, we note once again that the intention is to provide a simpler explanation of Algorithm 1 and to show the general relevance of maxflow-based techniques to loading scheduling, energy minimization, and computational sustainability. In the Discussion section, we comment on how, with a few additional insights, the same combinatorial arguments are applicable even if we use model A.

Notation

In this paper, we use the notation illustrated in Figure 3. We first identify L landmarks on the time axis, $\mathcal{L} = \{\ell_1, \ell_2, \ldots, \ell_L\}$, where $f(t)$ is discontinuous. We then divide the time axis into $(L+1)$ intervals, $\mathcal{I} = \{I_1, I_2, \ldots, I_{(L+1)}\}$, defined as follows: $I_1 = (-\infty, \ell_1]$, $I_k = (\ell_{(k-1)}, \ell_k]$ for $2 \leq k \leq L$, and $I_{(L+1)} = (\ell_L, +\infty)$. The infimum and supremum of an interval $(a, b]$ are defined as follows: $\inf\{(a, b]\} = a$ and $\sup\{(a, b]\} = b$. For technical reasons, we set ℓ_0 to $-\infty$ and $\ell_{(L+1)}$ to $+\infty$.



	X_0	X_c	X_h	X_d
X_0	0	1020	900	1380
X_c	-660	0	-90	600
X_h	-420	600	0	690
X_d	-1080	-60	-180	0

Figure 2: The left-hand side shows the STN that models an example application instance in the smart home domain. Each node represents an event. Each directed edge represents a simple temporal constraint specified using a lower and an upper bound. The bounds in the temporal constraints are in minutes. X_0 represents the beginning of the day, i.e., 12:00am. Each appliance requests and uses a certain amount of energy to work: the cooker uses 0.2kWh, the heater uses 1.3kWh, and the dishwasher uses 0.5kWh. The right-hand side shows the pairwise shortest path distances between the starting time points of all processes and X_0 in the distance graph of the STN. Each entry in the matrix represents the distance from the node in its row index to the node in its column index. X_c, X_h, and X_d are shorthand for the starting time points of the cooker, the heater, and the dishwasher, respectively.

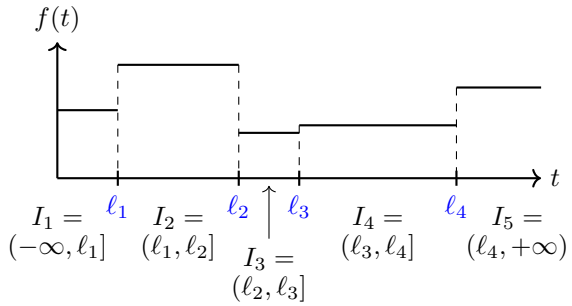


Figure 3: Illustrates the notation for landmarks and intervals in a piecewise constant function.

Algorithm for Cost Minimization

In this subsection, we provide a polynomial-time algorithm for finding the optimal schedule τ^* that is of minimum cost in a load scheduling problem on STNs.

Overview: We first introduce the idea of *activating* a process P_i in an interval I_j. Next, we prove that if activating a set of process-interval tuples leads to a *conflict*, then there exists a *minimal conflict* between some two of them. Therefore, the core minimal conflicts are only binary in nature and can be enumerated easily. Moreover, they can also be represented graphically as directed edges between nodes that correspond to process-interval tuples. In such a node-weighted "conflict graph", we show that the maximum weighted independent set corresponds to an optimal solution. Finally, we show that the conflict graph is a *partially ordered set* (POSET) on which the maximum weighted independent set can be computed efficiently in polynomial time using a maxflow algorithm on a bipartite graph.

Algorithm 1 provides the pseudocode of this algorithm; and the arguments for its correctness are presented below. We begin with the following well-known theorem about STNs and their distance graph representations (constructed in lines 2-7).

Theorem 1. *For a given STN S, $\Gamma(S) \neq \emptyset$ if and only if the distance graph $\mathcal{D}(S)$ does not contain any negative cost cycles. Furthermore, if $\mathcal{D}(S)$ does not contain any negative cost cycles, a consistent schedule of minimum makespan $\tau^m \in \Gamma(S)$ is given by $\tau^m(X_i) = -d(X_i, X_0)$.*

Proof. see (Dechter, Meiri, and Pearl 1991). □

Now, we first define the notion of the *activation* of a process P_i in an interval I_j. We then define the notions of a *conflict* and a *minimal conflict*.

Definition 2. *A process P_i is said to be activated in the interval I_j under a schedule τ if and only if $\tau(X^s_{P_i}) \in I_j$. For brevity, we also refer to this as the activation of the tuple $\langle P_i, I_j \rangle$ under τ.*

Definition 3. *A set $\{\langle P_{i_1}, I_{j_1}\rangle, \langle P_{i_2}, I_{j_2}\rangle, \ldots, \langle P_{i_M}, I_{j_M}\rangle\}$ of tuples constitutes a conflict if and only if there is no consistent τ under which all of the tuples can be activated.*

Definition 4. *A set $\{\langle P_{i_1}, I_{j_1}\rangle, \langle P_{i_2}, I_{j_2}\rangle, \ldots, \langle P_{i_M}, I_{j_M}\rangle\}$ of tuples constitutes a minimal conflict if and only if it constitutes a conflict and no proper subset of it constitutes a conflict.*

Lemma 5. *A set $\{\langle P_{i_1}, I_{j_1}\rangle, \langle P_{i_2}, I_{j_2}\rangle, \ldots, \langle P_{i_M}, I_{j_M}\rangle\}$ of tuples can be activated simultaneously if no subset of it constitutes a minimal conflict.*

Proof. By the definition of a conflict, a set of tuples can be simultaneously activated if and only if there is no subset of it that constitutes a conflict. Furthermore, every conflict has a minimal conflict, and therefore, a set of tuples can be simultaneously activated if and only if there is no subset of it that constitutes a minimal conflict. □

Our motivation for introducing Definitions 3 and 4, and proving Lemma 5 is to show that the sizes of all minimal conflicts are upper-bounded by a small constant, i.e., 2.

Theorem 6. *The size of a minimal conflict is ≤ 2.*

Proof. Suppose we try to simultaneously activate all tuples in the set $\{\langle P_{i_1}, I_{j_1}\rangle, \langle P_{i_2}, I_{j_2}\rangle, \ldots, \langle P_{i_M}, I_{j_M}\rangle\}$ under a schedule τ. In order to activate $\langle P_i, I_j\rangle$, we need to enforce the constraint that $\tau(X^s_{P_i}) \in I_j$. That is, we need

Algorithm 1: Shows a maxflow-based polynomial-time procedure for solving the load scheduling problem on STNs for the minimization of cost under a piecewise constant $f(t)$.

Function SOLVE-STN-LOAD-SCHEDULING

Input: An STN $S = \langle \mathcal{X}, \mathcal{E} \rangle$, where $\mathcal{X} = \{X_0, X_1, \ldots, X_N\}$ is the set of events and $\mathcal{E}$ is the set of simple temporal constraints between them;

Input: A set of processes $\mathcal{P}_S = \{P_1, P_2, \ldots, P_K\}$ with the starting time point $X^s_{P_i}$ and ending time point $X^e_{P_i}$ of each P_i included in $\mathcal{X}$;

Input: A piecewise constant $f(t)$ that models the dynamic unit price of energy;

Output: A consistent schedule $\tau^* \in \Gamma(S)$ that is of minimum cost;

• *Construct the distance graph $\mathcal{D}(S)$ on the nodes of S as follows:*

  for *each edge* $e = \langle X_i, X_j \rangle \in \mathcal{E}$ **do**

    Add the edge $\langle X_i, X_j \rangle$ annotated with $UB(e)$ to $\mathcal{D}(S)$;

    Add the edge $\langle X_j, X_i \rangle$ annotated with $-LB(e)$ to $\mathcal{D}(S)$;

  Compute the shortest path in $\mathcal{D}(S)$ between every pair of nodes;

  Let $d(X_i, X_j)$ be the length of the shortest path from X_i to X_j in $\mathcal{D}(S)$;

• *Name landmarks and intervals for $f(t)$:*

  Let the landmarks and intervals of $f(t)$ be $\mathcal{L} = \{\ell_1, \ell_2, \ldots, \ell_L\}$ and $\mathcal{I} = \{I_1, I_2, \ldots, I_{L+1}\}$, respectively;

• *Construct a POSET Λ as follows:*

  Let the elements of Λ be $\lambda_{\langle P_i, I_j \rangle}$ for all $P_i \in \mathcal{P}_S$ and for all $I_j \in \mathcal{I}$;

  Let $\lambda_{\langle P_u, I_a \rangle} \succ \lambda_{\langle P_v, I_b \rangle}$ if and only if $\ell_b + d(X^s_{P_v}, X^s_{P_u}) - \ell_{(a-1)} \leq 0$;

• *Construct a node-weighted directed acyclic graph G_Λ as follows:*

  Construct Y to be the set of nodes $\{y_{\langle P_i, I_j \rangle}\}$ of G_Λ that are in one-to-one correspondence with $\{\lambda_{\langle P_i, I_j \rangle} : d(X_0, X^s_{P_i}) - \ell_{(j-1)} > 0 \text{ and } \ell_j + d(X^s_{P_i}, X_0) \geq 0\}$;

  Construct a directed edge from $y_{\langle P_v, I_b \rangle}$ to $y_{\langle P_u, I_a \rangle}$ if and only if $\lambda_{\langle P_u, I_a \rangle} \succ \lambda_{\langle P_v, I_b \rangle}$;

  Let the weight on node $y_{\langle P_i, I_j \rangle}$ be $c_{ij} = [\sum_{i'} \max_{j'} W_{i'} \cdot f(\ell_{j'})] - W_i \cdot f(\ell_j) + 1$;

• *Compute the maximum weighted independent set Q_{G_Λ} of G_Λ as follows:*

  Construct a bipartite graph $B = \langle Y, Y', E \rangle$, where the nodes $\{y'_{\langle P_i, I_j \rangle}\}$ in Y' are in one-to-one correspondence with the nodes $\{y_{\langle P_i, I_j \rangle}\}$ in Y;

  Construct a directed edge from $y_{\langle P_u, I_a \rangle}$ to $y'_{\langle P_v, I_b \rangle}$ in B if and only if G_Λ has a directed edge from $y_{\langle P_u, I_a \rangle}$ to $y_{\langle P_v, I_b \rangle}$;

  Use the polynomial-time maxflow-based algorithm in (Goldberg and Tarjan 1988) to compute the maximum weighted independent set Q_B of B;

  $Q_{G_\Lambda} = \{y_{\langle P_i, I_j \rangle} : (y_{\langle P_i, I_j \rangle} \in Q_B) \wedge (y'_{\langle P_i, I_j \rangle} \in Q_B)\}$;

• *Construct a modified distance graph $\mathcal{D}'(S)$ on the nodes of S as follows:*

  Add all edges in $\mathcal{D}(S)$ to $\mathcal{D}'(S)$;

  for *each* $y_{\langle P_i, I_j \rangle} \in Q_{G_\Lambda}$ **do**

    Add the edge $\langle X_0, X^s_{P_i} \rangle$ annotated with ℓ_j to $\mathcal{D}'(S)$;

    Add the edge $\langle X^s_{P_i}, X_0 \rangle$ annotated with $-\ell_{(j-1)}$ to $\mathcal{D}'(S)$;

• *Compute and return the final schedule τ^* as follows:*

  Compute the shortest path in $\mathcal{D}'(S)$ from $X^s_{P_i}$ to X_0 for each $P_i \in \mathcal{P}_S$;

  Let $d'(X_i, X_j)$ be the length of the shortest path from X_i to X_j in $\mathcal{D}'(S)$;

  for *each* $P_i \in \mathcal{P}_S$ **do**

    Let $\tau^*(X^s_{P_i}) = -d'(X^s_{P_i}, X_0)$;

  return τ^*;

$\lim_{\epsilon \to 0} \inf\{I_j\} + |\epsilon| \leq \tau(X^s_{P_i}) \leq \sup\{I_j\}$, or equivalently, $\lim_{\epsilon \to 0} \ell_{(j-1)} + |\epsilon| \leq \tau(X^s_{P_i}) - \tau(X_0) \leq \ell_j$. In the distance graph representation, this requires us to add the outgoing edge $\langle X_0, X^s_{P_i} \rangle$ annotated with ℓ_j and the incoming edge $\langle X^s_{P_i}, X_0 \rangle$ annotated with $-(\ell_{(j-1)} + |\epsilon|)$.[2] Therefore, to activate all tuples $\langle P_{i_1}, I_{j_1} \rangle, \langle P_{i_2}, I_{j_2} \rangle, \ldots, \langle P_{i_M}, I_{j_M} \rangle$, we need to add the following edges to the distance graph $\mathcal{D}(S)$ without creating a negative cost cycle: edges $\langle X_0, X^s_{P_{i_r}} \rangle$

[2] Here, "outgoing" and "incoming" are with respect to X_0.

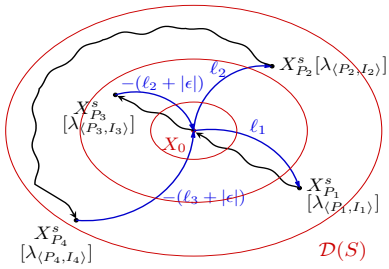


Figure 4: Illustrates the arguments used in the proofs of Theorem 6 and Lemmas 7 and 8. The red curves indicate the distance graph, $\mathcal{D}(S)$; the blue curves indicate the activation edges; and the black curves indicate shortest paths in $\mathcal{D}(S)$.

annotated with ℓ_{j_r} and edges $\langle X^s_{P_{i_r}}, X_0\rangle$ annotated with $-(\ell_{(j_r-1)}+|\epsilon|)$ for all $1 \leq r \leq M$. Let's refer to these edges as "activation" edges. From Theorem 1, we know that for a consistent STN S, there are no negative cost cycles in $\mathcal{D}(S)$. If a conflict occurs in activating the M tuples, then a negative cost cycle is newly created with the addition of the activation edges. This means that any such newly created negative cost cycle must involve at least one of the activation edges. However, since every activation edge involves X_0 as one of its end points, the negative cost cycle can involve at most two such activation edges. This is because any fundamental cycle in a graph can involve any node at most once. This proves the theorem; and Figure 4 illustrates this argument. □

From Theorem 6, we know that every minimal conflict is of size either 1 or 2. We correspondingly refer to them as *unary* and *binary* minimal conflicts. In lines 10-12, we construct a POSET $\Lambda = \{\lambda_{\langle P_i,I_j\rangle}\}$, where $\lambda_{\langle P_i,I_j\rangle}$ represents the activation of process P_i in the interval I_j and $\lambda_{\langle P_u,I_a\rangle} \succ \lambda_{\langle P_v,I_b\rangle}$ represents a binary minimal conflict between $\langle P_u, I_a\rangle$ and $\langle P_v, I_b\rangle$.

Lemma 7. *$\lambda_{\langle P_i,I_j\rangle}$ represents a unary minimal conflict if and only if $d(X_0, X^s_{P_i}) - \ell_{(j-1)} \leq 0$ or $\ell_j + d(X^s_{P_i}, X_0) < 0$.*

Proof. We know that $\lambda_{\langle P_i,I_j\rangle}$ represents the addition of two edges: $\langle X_0, X^s_{P_i}\rangle$ annotated with ℓ_j (edge 1) and $\langle X^s_{P_i}, X_0\rangle$ annotated with $-(\ell_{(j-1)}+|\epsilon|)$ (edge 2). A unary conflict is created in one of the following two cases: (a) edge 1 is the only activation edge in a newly created negative cost cycle that happens when $\ell_j + d(X^s_{P_i}, X_0) < 0$; or (b) edge 2 is the only activation edge in a newly created negative cost cycle that happens when $d(X_0, X^s_{P_i}) - \ell_{(j-1)} \leq 0$ (because $\epsilon \to 0$). Figure 4 illustrates these two cases. □

Lemma 8. *$\lambda_{\langle P_u,I_a\rangle}$ and $\lambda_{\langle P_v,I_b\rangle}$ together represent an asymmetric binary minimal conflict, denoted by $\lambda_{\langle P_u,I_a\rangle} \succ \lambda_{\langle P_v,I_b\rangle}$ without loss of generality, if and only if $\ell_b + d(X^s_{P_v}, X^s_{P_u}) - \ell_{(a-1)} \leq 0$.*

Proof. We know that $\lambda_{\langle P_u,I_a\rangle}$ represents the addition of two edges: $\langle X_0, X^s_{P_u}\rangle$ annotated with ℓ_a (edge 1) and $\langle X^s_{P_u}, X_0\rangle$ annotated with $-(\ell_{(a-1)}+|\epsilon|)$ (edge 2). Similarly, $\lambda_{\langle P_v,I_b\rangle}$ represents the addition of two edges: $\langle X_0, X^s_{P_v}\rangle$ annotated with ℓ_b (edge 3) and $\langle X^s_{P_v}, X_0\rangle$ annotated with $-(\ell_{(b-1)}+|\epsilon|)$ (edge 4). A binary minimal conflict involves exactly one incoming activation edge and one outgoing activation edge (with respect to X_0) in a newly created negative cost cycle. Without loss of generality, let edge 2 be the incoming edge and edge 3 be the outgoing edge. They together participate in a negative cost cycle if and only if $\ell_b + d(X^s_{P_v}, X^s_{P_u}) - \ell_{(a-1)} \leq 0$ (because $\epsilon \to 0$). Figure 4 illustrates a binary minimal conflict. □

By convention and without loss of generality, $\lambda_{\langle P_u,I_a\rangle} \succ \lambda_{\langle P_v,I_b\rangle}$ is used to denote a binary minimal conflict when the lefthand side contributes the incoming edge and the righthand side contributes the outgoing edge (with respect to X_0).

While the number of conflicts might be exponential in the number of processes, K, and the number of landmarks, L, and therefore unwieldy to enumerate explicitly, we can exploit the upper bound on the size of any minimal conflict to efficiently enumerate all of the minimal conflicts instead (lines 10-12 of Algorithm 1).

Theorem 9. *The number of minimal conflicts is only quadratic in the number of processes, K, and the number of landmarks, L.*

Proof. The number of $\lambda_{\langle P_i,I_j\rangle}$'s is equal to $K \cdot (L+1)$. From Theorem 6, we know that all minimal conflicts are at most binary. Therefore, the maximum number of minimal conflicts is $O(K^2L^2)$. □

We now prove some special properties of the $\succ$ relation, culminating in its induction of a valid POSET.

Lemma 10. *For any process P_i and any $1 \leq k < j \leq (L+1)$, $\lambda_{\langle P_i,I_j\rangle} \succ \lambda_{\langle P_i,I_k\rangle}$.*

Proof. We know that $\lambda_{\langle P_i,I_j\rangle}$ requires the addition of two edges: $\langle X_0, X^s_{P_i}\rangle$ annotated with ℓ_j (edge 1) and $\langle X^s_{P_i}, X_0\rangle$ annotated with $-(\ell_{(j-1)}+|\epsilon|)$ (edge 2). Similarly, $\lambda_{\langle P_i,I_k\rangle}$ requires the addition of two edges: $\langle X_0, X^s_{P_i}\rangle$ annotated with ℓ_k (edge 3) and $\langle X^s_{P_i}, X_0\rangle$ annotated with $-(\ell_{(k-1)}+|\epsilon|)$ (edge 4). If $j > k$, we know that $\ell_{(j-1)} + |\epsilon| > \ell_k$. Therefore, edge 2 and edge 3 create a negative cost cycle leading to the conflict $\lambda_{\langle P_i,I_j\rangle} \succ \lambda_{\langle P_i,I_k\rangle}$. □

Lemma 11. *The binary relation $\succ$ is transitive.*

Proof. Suppose we have $\lambda_{\langle P_u,I_a\rangle} \succ \lambda_{\langle P_v,I_b\rangle}$ and $\lambda_{\langle P_v,I_b\rangle} \succ \lambda_{\langle P_w,I_c\rangle}$. We show that $\lambda_{\langle P_u,I_a\rangle} \succ \lambda_{\langle P_w,I_c\rangle}$. From Lemma 8, we know that $\ell_b + d(X^s_{P_v}, X^s_{P_u}) - \ell_{(a-1)} \leq 0$, and $\ell_c + d(X^s_{P_w}, X^s_{P_v}) - \ell_{(b-1)} \leq 0$. Adding these two inequalities, we get $[\ell_b - \ell_{(b-1)}] + [d(X^s_{P_v}, X^s_{P_u}) + d(X^s_{P_w}, X^s_{P_v})] + \ell_c - \ell_{(a-1)} \leq 0$. We also know that $\ell_b \geq \ell_{(b-1)}$, and $d(X^s_{P_w}, X^s_{P_v}) + d(X^s_{P_v}, X^s_{P_u}) \geq d(X^s_{P_w}, X^s_{P_u})$, by triangle inequality on shortest path distances. Therefore, $\ell_c + d(X^s_{P_w}, X^s_{P_u}) - \ell_{(a-1)} \leq 0$. From Lemma 8, this means $\lambda_{\langle P_u,I_a\rangle} \succ \lambda_{\langle P_w,I_c\rangle}$. □

Lemma 12. *The binary relation $\succ$ is acyclic.*

Proof. Suppose we have a cycle $\lambda_{\langle P_{i_1},I_{j_1}\rangle} \succ \lambda_{\langle P_{i_2},I_{j_2}\rangle} \succ \ldots \lambda_{\langle P_{i_M},I_{j_M}\rangle} \succ \lambda_{\langle P_{i_1},I_{j_1}\rangle}$. From Lemma 11, we know that $\lambda_{\langle P_{i_1},I_{j_1}\rangle} \succ \lambda_{\langle P_{i_1},I_{j_1}\rangle}$. From Lemma 8, this means that $\ell_{j_1} + d(X^s_{P_{i_1}}, X^s_{P_{i_1}}) - \ell_{(j_1-1)} \leq 0$, or equivalently, $\ell_{(j_1-1)} - \ell_{j_1} \geq d(X^s_{P_{i_1}}, X^s_{P_{i_1}})$. This is a contradiction because $\ell_{j_1} > \ell_{(j_1-1)}$ and $d(X^s_{P_{i_1}}, X^s_{P_{i_1}}) = 0$. Therefore, $\succ$ is acyclic. □

Theorem 13. *The binary relation $\succeq$ defined as follows forms a POSET: $\lambda_{\langle P_u,I_a\rangle} \succeq \lambda_{\langle P_v,I_b\rangle}$ if and only if $\lambda_{\langle P_u,I_a\rangle} \succ \lambda_{\langle P_v,I_b\rangle}$ or $(u,a) = (v,b)$.*

Proof. By definition, $\succeq$ is reflexive. By Lemma 11, it is transitive, and by Lemma 12, it is anti-symmetric. Therefore, $\succeq$ forms a POSET on the elements of Λ. □

We note that although Λ is a POSET under the relation $\succeq$, the real benefit of recognizing its properties is to represent the elements of Λ under the relation $\succ$ using a directed acyclic graph G_Λ. Lines 13-15 of Algorithm 1 construct G_Λ, using which it is easy to generalize the characterization of a consistent solution to the characterization of an optimal solution of minimum cost.

Lemma 14. *The STN S with the set of processes $\mathcal{P}(S) = \{P_1, P_2, \ldots, P_K\}$ is consistent if and only if the size of the maximum independent set in G_Λ is K.*

Proof. From Theorem 6, we know that all minimal conflicts are either unary or binary. Line 14 of Algorithm 1 constructs a set of nodes $Y = \{y_{\langle P_i,I_j\rangle}\}$ for G_Λ that are in one-to-one correspondence with only those elements of Λ that do not constitute a unary minimal conflict (Lemma 7). Line 15 represents all the binary minimal conflicts using directed edges between the nodes in Y. An independent set of G_Λ therefore represents a collection of $\langle P_i, I_j\rangle$ tuples that can be activated simultaneously. Furthermore, from Lemma 10, any process P_i can be activated in at most one interval. Since every process P_i needs to be activated in some interval, exactly one interval is activated for each process in a consistent STN. Equivalently, this means that the size of the maximum independent set of G_Λ is K, where each node of the maximum independent set corresponds to a different process. □

Theorem 15. *Suppose the weight on node $y_{\langle P_i,I_j\rangle}$ is $c_{ij} = [\sum_{i'} \max_{j'} W_{i'} \cdot f(\ell_{j'})] - W_i \cdot f(\ell_j) + 1$, where $1 \leq i \leq K$ and $1 \leq j \leq (L+1)$. Then, a solution with minimum cost corresponds to the activation of a maximum weighted independent set Q_{G_Λ} of G_Λ.*

Proof. We first prove that the activation of Q_{G_Λ} corresponds to a consistent solution. By virtue of Lemma 14, it suffices to prove that Q_{G_Λ} is of cardinality K. From Lemma 10, the cardinality of Q_{G_Λ} is $\leq K$. Suppose the cardinality of Q_{G_Λ} is $< K$. Then the total weight of Q_{G_Λ} is $\leq (K-1)[\sum_{i'} \max_{j'} W_{i'} \cdot f(\ell_{j'})] + (K-1)$. From Lemma 14, for a consistent STN S, there exists an independent set of cardinality K. Let this independent set be $T_{G_\Lambda} = \{y_{\langle P_1,I_{r_1}\rangle}, y_{\langle P_2,I_{r_2}\rangle}, \ldots, y_{\langle P_K,I_{r_K}\rangle}\}$, since, from Lemma 10, exactly one interval should be activated for each process. The total weight of $T_{G_\Lambda} = K[\sum_{i'} \max_{j'} W_{i'} \cdot f(\ell_{j'})] - \sum_i [W_i \cdot f(\ell_{r_i})] + K \geq (K-1)[\sum_{i'} \max_{j'} W_{i'} \cdot f(\ell_{j'})] + K$. This contradicts that Q_{G_Λ}, whose total weight is $\leq (K-1)[\sum_{i'} \max_{j'} W_{i'} \cdot f(\ell_{j'})] + (K-1)$, is the maximum weighted independent set, hence proving that the cardinality of Q_{G_Λ} is exactly K. We now prove that the activation of Q_{G_Λ} also corresponds to a solution with minimum cost. Consider any independent set of cardinality K, $T_{G_\Lambda} = \{y_{\langle P_1,I_{r_1}\rangle}, y_{\langle P_2,I_{r_2}\rangle}, \ldots, y_{\langle P_K,I_{r_K}\rangle}\}$. The quality of any solution for the STN induced by activating T_{G_Λ} is equal to its total weight, given by $\{[K \sum_{i'} \max_{j'} W_{i'} \cdot f(\ell_{j'})] + K\} - \sum_i [W_i \cdot f(\ell_{r_i})]$. Since $[K \sum_{i'} \max_{j'} W_{i'} \cdot f(\ell_{j'})] + K$ is constant for all solutions, the maximization of the total weight is equivalent to the minimization of the cost $\sum_i [W_i \cdot f(\ell_{r_i})]$. □

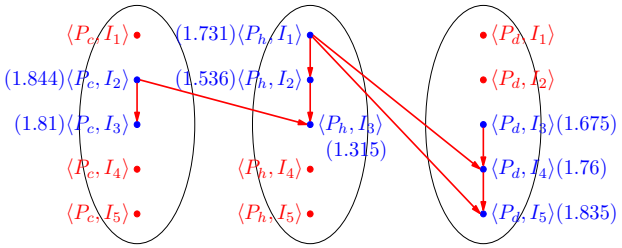


Figure 5: Shows the construction of the graph G_Λ for the running example from Figure 2 and the pricing from Figure 1. Here, the subscripts c, h, and d correspond to the cooker, the heater, and the dishwasher, respectively. The red nodes indicate unary minimal conflicts and the red edges indicate binary minimal conflicts. Each $\langle P, I\rangle$ annotation is shorthand for $y_{\langle P,I\rangle}$, represented using a blue node in G_Λ. Each blue node is also annotated with its weight calculated in line 16. For example, the weight on $\langle P_c, I_2\rangle$ is set to 1.844, because "$[\sum_{i'} \max_{j'} W_{i'} \cdot f(\ell_{j'})] - W_i \cdot f(\ell_j) + 1$" is equal to $(0.2 \cdot 0.45 + 1.3 \cdot 0.45 + 0.5 \cdot 0.45) - 0.2 \cdot 0.28 + 1 = 1.844$. The maximum weighted independent set is the choice of $\langle P_c, I_2\rangle$, $\langle P_h, I_1\rangle$, and $\langle P_d, I_3\rangle$. Activating these tuples yields a consistent solution of minimum cost.

We note that the weights on the nodes of G_Λ are ≥ 0. After they are set according to the requirement of Theorem 15 (line 16), the maximum weighted independent set in G_Λ is computed (lines 17-21) using a polynomial-time maxflow procedure on a staged bipartite graph (Goldberg and Tarjan 1988). Although the maximum weighted independent set problem is NP-hard in general, it is tractable for directed acyclic graphs that exhibit the transitive property (Golumbic 2004). Lines 22-26 activate Q_{G_Λ} by adding new activation edges to the distance graph. Using this modified distance graph, lines 27-32 return a schedule τ^* with minimum cost. Figure 5 illustrates the mathematical transformations used in the correctness arguments of the algorithm for the running example from Figure 2.

The time complexity of Algorithm 1 is dominated by lines 6 and 20. Line 6 is of complexity $O(K^2 N|\mathcal{E}|)$, and line 20 is of complexity $O(K^{2.5})$ (Goldberg and Tarjan 1988). Since $N > K$, the overall time complexity is $O(K^2 N|\mathcal{E}|)$.

Algorithm 2: Shows a quasi binary search algorithm for trading off makespan minimization against cost minimization within a suboptimality factor.

1 **Function** SOLVE-STN-LOAD-MAKESPAN
  Input: A load scheduling problem on an STN with input parameters $S = \langle \mathcal{X}, \mathcal{E} \rangle$, $\mathcal{P}_S = \{P_1, P_2, \ldots, P_K\}$ and $f(t)$;
  Input: A suboptimality factor $\gamma \geq 1$ for cost minimization in the trade-off against makespan minimization;
  Output: A consistent schedule τ^* that is of minimum makespan among all consistent schedules of cost less than or equal to γ times the minimum cost;
2 • *Initialize:*
3 $\tau =$ SOLVE-STN-LOAD-SCHEDULING $(S, \mathcal{P}_S, f(t))$;
4 Let c^* be the cost of τ and let μ be the makespan of τ;
5 Set the lower bound for makespan $lbms$ to 0 and the upper bound for makespan $ubms$ to μ;
6 • *Conduct quasi binary search:*
7 **while** $(ubms - lbms)$ *is not sufficiently small* **do**
8 $m = (lbms + ubms)/2$;
9 **for** *each process* $P_i \in \{P_1, P_2, \ldots, P_K\}$ **do**
10 Add the simple temporal constraint $\langle X_0, X^e_{P_i} \rangle$ annotated with $[0, m]$ to S;
11 **if** S *is consistent* **then**
12 $\tau =$ SOLVE-STN-LOAD-SCHEDULING $(S, \mathcal{P}_S, f(t))$;
13 Let c be the cost of τ and ν be the makespan of τ;
14 **if** $c \leq \gamma \cdot c^*$ **then**
15 $ubms = \nu$;
16 **else**
17 $lbms = m$;
18 **else**
19 $lbms = m$;
20 • *Return the final solution:*
21 return $\tau^* = \tau$;

Algorithm for Makespan vs Cost Minimization

While Algorithm 1 minimizes the cost for a given load scheduling problem, it is oblivious to makespan minimization. In many real-world applications, however, makespan minimization constitutes an important secondary objective. In this subsection, we provide a polynomial-time quasi binary search algorithm, Algorithm 2, for trading off makespan minimization against cost minimization within a user-specified suboptimality factor γ.

Algorithm 2 is based on the critical observation that enforcing a makespan of m in a load scheduling problem can be done using additional simple temporal constraints of the form $\langle X_0, X^e_{P_i} \rangle$ annotated with $[0, m]$ for all P_i (line 10). Binary search is carried out in lines 6-19 to find the minimum value of makespan that yields a solution of cost within the user-specified suboptimality bound. The binary search is optimized (line 15) under the following note. When a makespan constraint of $m = (lbms + ubms)/2$ is imposed, if a consistent schedule τ can be successfully found within the suboptimality bound, $ubms$ can be tightened to the makespan of τ for the next iteration (instead of to m as in regular binary search). We refer to this optimized version as quasi binary search.

The running time of Algorithm 2 is $O(\log_2 \mu \cdot K^2 N |\mathcal{E}|)$. Here, $K^2 N |\mathcal{E}|$ represents the running time of Algorithm 1, and $\log_2 \mu$ represents the number of times Algorithm 1 is called in the binary search. Although μ depends on the numerical values in the problem instance, $\log_2 \mu$ is only linear in the size of the bit representation of these numbers. Therefore, in polynomial time, we can find a solution of minimum makespan that is within the user-specified suboptimality bound γ with respect to cost.

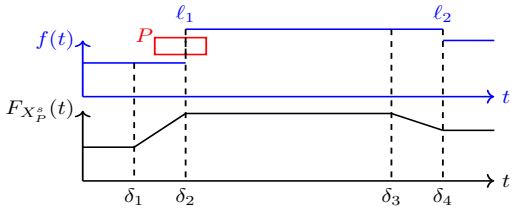


Figure 6: Shows the derivation of $F_{X^s_P}(t)$ for process P under a piecewise constant $f(t)$. $F_{X^s_P}(t)$ represents the cost of scheduling P to start at time t. The result is a piecewise linear function, where discretization is required only within the intervals $(\delta_1, \delta_2]$ and $(\delta_3, \delta_4]$ (instead of the entire timeline).

Discussion

In this section, we elaborate on some possible extensions to the load scheduling problem studied so far. We first discuss its variant for model A. Specifically, we comment on the applicability of our techniques with discretization of time, but only in certain intervals. We then consider the load scheduling problem under the cost function $f(t, u(t))$, where the unit cost of energy depends on time t as well as the total demand $u(t)$ at that time.

Figure 6 shows the cost of scheduling the starting time point X^s_P of process P at time t using model A. Here, the process P is of duration q and consumes energy at the rate of w watts. The cost curve $F_{X^s_P}(t)$ is then the "convolution" of $f(t)$ with the duration of the process q. It is therefore different for different processes and is piecewise linear.

Ignoring the second issue, i.e., assuming piecewise constant cost functions, the first issue can be easily addressed using the following simple modification to line 16 of Algorithm 1: $c_{ij} = [\sum_{i'} \max_{j'} F_{X^s_{P_{i'}}}(\ell_{j'})] - F_{X^s_{P_i}}(\ell_j) + 1$. In fact, it is relatively easy to prove generalized versions of our lemmas and theorems, as alluded to in (Kumar 2004).

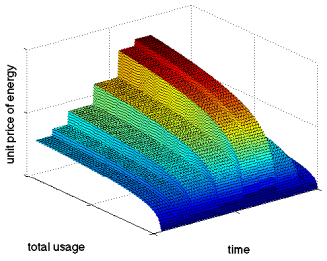


Figure 7: Illustrates a general cost function $f(t, u(t))$. Here, the cost function is piecewise constant with respect to time, and is concave with respect to the total demand at any fixed time. This case is conjectured to be tractable.

The second issue can be handled by discretization of time to make the cost curves piecewise constant. Based on the reasonable assumption that process durations are much shorter than the spacing between the landmarks of $f(t)$, we note here that discretization is necessary only in narrow intervals of time near the landmarks of $f(t)$, where the cost curves are otherwise piecewise linear. This yields huge savings in comparison to a discretization of the entire time line that is used in many other works, such as in (Goudarzi, Hatami, and Pedram 2011) under the name of a *slotted time* model. Figure 6 illustrates this and demonstrates that the granularity of discretization depends on q as well as the difference in values at discontinuities of $f(t)$. Since this difference is usually small, discretization is not very expensive in our framework. In addition, we conjecture that piecewise linear cost curves are also amenable to polynomial-time maxflow-based algorithms; and settling this conjecture is part of our future work.

We also consider a generalized cost function $f(t, u(t))$ as shown in Figure 7. In this model, the energy provider could either encourage or penalize "buying in bulk" based on the balance of supply and demand.

Theorem 16. *If $f(t, u(t))$ has a convex dependency on $u(t)$, the load scheduling problem is* NP*-hard.*

Proof. In the decision version of the bin packing problem, given n items of sizes $a_1, a_2, \ldots, a_n$ and B bins each of capacity V, it is NP-hard to determine whether all the items can be incorporated into the B bins without exceeding the capacity constraints. It suffices to show that this problem is reducible to a special case of the load scheduling problem where the cost function $f(t, u(t))$ has a convex dependency on $u(t)$. Let the cost function be independent of t and depend only on the total demand u, such that $f(u) = 0$ if $u \leq H$; and $f(u) = +\infty$ otherwise. We create n processes $P_1, P_2, \ldots, P_n$ where the power requirement of process P_i is equal to a_i. Each process is of unit duration, and is allowed to execute in the time window $[0, B]$. The threshold H is set to the capacity V. Consider the B bins to be represented in the B intervals $[0, 1], [1, 2], \ldots, [B-1, B]$. Under this construction, a satisfying solution for the bin packing problem exists if and only if a satisfying solution for the load scheduling problem exists. □

If $f(t, u(t))$ has a linear dependency on $u(t)$, it can be decomposed to $\sum_{P_i} f(t, u_{P_i}(t))$. Here, $u_{P_i}(t)$ is the demand of process P_i at time t, and is equal to w_i in model A, which we conjectured above to be tractable based on the algorithm presented in this paper. We now make a stronger conjecture that when the dependency of $f(t, u(t))$ on $u(t)$ is concave, i.e., when "buying in bulk" is encouraged, the load scheduling problem remains tractable based on arguments of *submodularity* (Lovász 1983); and settling this stronger conjecture is also part of our future work.

Related Work

Load scheduling problems have been extensively studied in the smart home and smart grid domains. Typically, the goal is to minimize the cost of energy consumption when prices fluctuate with time. In (Agnetis et al. 2013), a Mixed Integer Program (MIP) is used for trading off household energy cost against loss of comfort. In (Antonopoulos, Kapsalis, and Hadellis 2012), exhaustive search is used for optimal scheduling. (Pipattanasomporn, Kuzlu, and Rahman 2012) proposes an algorithm for keeping power consumption below certain levels based on preset priorities of processes. (Zhao et al. 2013) provides another power scheduling method using a genetic algorithm. (Wang et al. 2013) presents a traversal-and-pruning algorithm for load scheduling in household applications. In real-time pricing environments, (Mohsenian-Rad and Leon-Garcia 2010) interleaves price prediction and load control. Similar problems in smart homes and smart grids with user preferences and dynamic pricing are solved using reformulations to variants of the knapsack problem in (Rasheed et al. 2016) and (Sianaki, Hussain, and Tabesh 2010). (Qayyum et al. 2015) provides a survey of related problems and techniques including MIP and constraint programming methods such as branch-and-bound. (Tabakhi et al. 2017) studies smart home device scheduling using the framework of distributed constraint optimization problems.

When the load scheduling problem faces the hard constraint of the total power consumption having to be below a cap at all times, the problem is NP-hard. Such variants of the load scheduling problem arise in job shop scheduling as well as in the HPC domain. In job shop scheduling, processes that compete for the same resource cannot have overlapping executions (Smith and Cheng 1993). This can be enforced by associating a unit cost with each process and a global cap of unit cost as well. In the HPC domain, the cap represents computational and cooling resources. The Least Slack First (LSF) algorithm is used in (Barker et al. 2012) to flatten the peak electrical power demand in smart homes.

Many other heuristics have been developed for several other NP-hard variants of the load scheduling problem with a cap. For example, (Goudarzi, Hatami, and Pedram 2011) presents a rank-based heuristic and a force directed-based heuristic. (Wallace et al. 2016) presents a data-driven scheduling approach for power management on HPC systems. (Meng et al. 2015) presents strategies for jointly reducing communication and cooling costs. (Sarood 2014) tries to optimize performance under thermal and power constraints;

and (Bodas et al. 2014) provides a power-aware scheduler to limit power consumption within a budget.

Resource-constrained scheduling has also been studied in (Sidor et al. 2016) with a smart home application under the name of Time Resource Networks. Here, MIP and constraint programming techniques are proposed as the solution methods. Robust resource allocation problems with uncertainty and makespan minimization is studied in (Wiesemann, Kuhn, and Rustem 2011). Resource envelope problems that provide upper and lower bounds on the resource utilization of consistent schedules of STNs are presented in (Kumar 2003). (Oddi et al. 2010) applies Iterative Flattening Search, a meta-heuristic strategy, to resource-constrained scheduling problems. STNs with preferences on individual variables have been studied in (Kumar 2004) and (Kumar, Cirillo, and Koenig 2013). General temporal reasoning problems with uncertainty and/or preferences have been studied in (Yorke-Smith, Venable, and Rossi 2003; Morris et al. 2004; Terenziani, Andolina, and Piovesan 2017).

Conclusions and Future Work

In this paper, we studied load scheduling problems on STNs—that represent various processes and constraints between their execution times—under dynamic pricing of resources (energy). We provided a polynomial-time algorithm that finds a solution of minimum cost for the load scheduling problem on STNs when the unit price of energy is a piecewise constant function of time. Our polynomial-time algorithm was based on the idea of reducing this problem to the problem of computing the maximum weighed independent set in a POSET, which in turn was solved using a maxflow procedure. We then used the polynomial-time algorithm in a quasi binary search procedure to trade off makespan minimization against cost minimization. Our algorithms have important applications in many real-world domains including efficient appliance scheduling in the smart home and energy minimization in the smart grid domains, where currently inefficient or suboptimal algorithms are being applied. We then studied the dependency of the unit price of energy on time as well as the total energy demand at that time. This led to a further characterization of tractable, NP-hard, and conjectured tractable cases of load scheduling problems.

There are many avenues for future work. One is to identify richer tractable cases, especially for model A and/or when the unit price of energy is a function that is piecewise constant in time and is concave in the total energy demand at any given time. We are also interested in finding resource-envelope-based heuristics for the NP-hard cases (Kumar 2003). In addition, we would also like to apply our algorithms to real-world domains in collaboration with researchers in electrical and civil engineering.

Acknowledgment

This material is based upon work supported by the Air Force Research Laboratory (AFRL) and the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR0011-15-C-0138. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the official views or policies of the Department of Defense or the U.S. Government.

References

Agnetis, A.; de Pascale, G.; Detti, P.; and Vicino, A. 2013. Load scheduling for household energy consumption optimization. *IEEE Transactions on Smart Grid* 4(4):2364–2373.

Antonopoulos, C. P.; Kapsalis, V.; and Hadellis, L. 2012. Optimal scheduling of smart homes' appliances for the minimization of energy cost under dynamic pricing. In *Proceedings of the IEEE Conference on Emerging Technologies & Factory Automation (ETFA)*, 1–4. IEEE.

Barker, S.; Mishra, A.; Irwin, D.; Shenoy, P.; and Albrecht, J. 2012. Smartcap: Flattening peak electricity demand in smart homes. In *Proceedings of the IEEE Conference on Pervasive Computing and Communications (PerCom)*, 67–75. IEEE.

Bodas, D.; Song, J.; Rajappa, M.; and Hoffman, A. 2014. Simple power-aware scheduler to limit power consumption by hpc system within a budget. In *Proceedings of the 2nd International Workshop on Energy Efficient Supercomputing*, 21–30. IEEE Press.

Borenstein, S. 2005. The long-run efficiency of real-time electricity pricing. *The Energy Journal* 93–116.

Dechter, R.; Meiri, I.; and Pearl, J. 1991. Temporal constraint networks. *Artificial Intelligence* 49(1-3):61–95.

Goldberg, A. V., and Tarjan, R. E. 1988. A new approach to the maximum-flow problem. *Journal of the ACM (JACM)* 35(4).

Golumbic, M. C. 2004. *Algorithmic graph theory and perfect graphs*, volume 57. Elsevier.

Goudarzi, H.; Hatami, S.; and Pedram, M. 2011. Demand-side load scheduling incentivized by dynamic energy prices. In *IEEE International Conference on Smart Grid Communications*.

Ji, M.; He, Y.; and Cheng, T. E. 2007. Single-machine scheduling with periodic maintenance to minimize makespan. *Computers & Operations Research* 34(6):1764–1770.

Knight, S.; Rabideau, G.; Chien, S.; Engelhardt, B.; and Sherwood, R. 2001. Casper: Space exploration through continuous planning. *IEEE Intelligent Systems* 16(5):70–75.

Kumar, T. K. S.; Cirillo, M.; and Koenig, S. 2013. Simple temporal problems with taboo regions. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence (AAAI)*.

Kumar, T. K. S. 2003. Incremental computation of resource-envelopes in producer-consumer models. In *Proceedings of the 9th International Conference on Principles and Practice of Constraint Programming (CP)*.

Kumar, T. K. S. 2004. A polynomial-time algorithm for simple temporal problems with piecewise constant domain preference functions. In *Proceedings of the 19th National Conference on Artificial Intelligence (AAAI)*.

Lovász, L. 1983. Submodular functions and convexity. In *Mathematical Programming The State of the Art*. Springer. 235–257.

Meng, J.; McCauley, S.; Kaplan, F.; Leung, V. J.; and Coskun, A. K. 2015. Simulation and optimization of hpc job allocation

for jointly reducing communication and cooling costs. *Sustainable Computing: Informatics and Systems* 6.

Mohsenian-Rad, A.-H., and Leon-Garcia, A. 2010. Optimal residential load control with price prediction in real-time electricity pricing environments. *IEEE transactions on Smart Grid* 1(2).

Morris, P.; Morris, R.; Khatib, L.; Ramakrishnan, S.; and Bachmann, A. 2004. Strategies for global optimization of temporal preferences. In *Proceedings of the 10th International Conference on Principles and Practice of Constraint Programming (CP)*. Springer.

Muscettola, N. 2004. Incremental maximum flows for fast envelope computation. In *Proceedings of the 14th International Conference on Automated Planning and Scheduling (ICAPS)*.

Nattaf, M.; Artigues, C.; and Lopez, P. 2017. Cumulative scheduling with variable task profiles and concave piecewise linear processing rate functions. *Constraints* 1–18.

Oddi, A.; Cesta, A.; Policella, N.; and Smith, S. F. 2010. Iterative flattening search for resource constrained scheduling. *Journal of Intelligent Manufacturing* 21(1):17–30.

Pipattanasomporn, M.; Kuzlu, M.; and Rahman, S. 2012. An algorithm for intelligent home energy management and demand response analysis. *IEEE Transactions on Smart Grid* 3(4).

Planken, L.; de Weerdt, M.; and van der Krogt, R. 2008. P^3C: a new algorithm for the simple temporal problem. In *Proceedings of the 18th International Conference on Automated Planning and Scheduling (ICAPS)*, 256–263.

Qayyum, F.; Naeem, M.; Khwaja, A. S.; Anpalagan, A.; Guan, L.; and Venkatesh, B. 2015. Appliance scheduling optimization in smart home networks. *IEEE Access* 3:2176–2190.

Rasheed, M. B.; Javaid, N.; Ahmad, A.; Jamil, M.; Khan, Z. A.; Qasim, U.; and Alrajeh, N. 2016. Energy optimization in smart homes using customer preference and dynamic pricing. *Energies*.

Sarood, O. 2014. *Optimizing performance under thermal and power constraints for HPC data centers*. Ph.D. Dissertation, University of Illinois at Urbana-Champaign.

Sianaki, O. A.; Hussain, O.; and Tabesh, A. R. 2010. A knapsack problem approach for achieving efficient energy consumption in smart grid for endusers' life style. In *Proceedings of the IEEE Conference on Innovative Technologies for an Efficient and Reliable Electricity Supply (CITRES)*, 159–164. IEEE.

Sidor, S.; Yu, P.; Fang, C.; and Williams, B. C. 2016. Time resource networks. *CoRR* abs/1602.03203.

Smith, S. F., and Cheng, C.-C. 1993. Slack-based heuristics for constraint satisfaction scheduling. In *Proceedings of the 11th National Conference on Artificial Intelligence (AAAI)*, 139–144.

Southern California Edison. 2017. *Time-Of-Use Residential Rate Plans*. `https://www.sce.com/wps/portal/home/residential/rates/Time-Of-Use-Residential-Rate-Plans`.

Tabakhi, A. M.; Le, T.; Fioretto, F.; and Yeoh, W. 2017. Preference elicitation for dcops. In *Proceedings of the 23rd International Conference on Principles and Practice of Constraint Programming (CP)*, 278–296.

Terenziani, P.; Andolina, A.; and Piovesan, L. 2017. Managing temporal constraints with preferences: Representation, reasoning, and querying. *IEEE Transactions on Knowledge and Data Engineering*.

U.S. Department of Energy. 2006. Benefits of demand response in electricity markets and recommendations for achieving them. *A report to the United States Congress*.

Wallace, S.; Yang, X.; Vishwanath, V.; Allcock, W. E.; Coghlan, S.; Papka, M. E.; and Lan, Z. 2016. A data driven scheduling approach for power management on hpc systems. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 656–666. IEEE.

Wang, C.; Zhou, Y.; Wang, J.; and Peng, P. 2013. A novel traversal-and-pruning algorithm for household load scheduling. *Applied energy* 102:1430–1438.

Wiesemann, W.; Kuhn, D.; and Rustem, B. 2011. Robust resource allocations in temporal networks. *Mathematical programming*.

Xiong, Y.; Sadeh, N. M.; and Sycara, K. P. 1992. Intelligent backtracking techniques for job shop scheduling. In *Proceedings of the 3rd International Conference on Principles of Knowledge Representation and Reasoning (KR)*, 14–23.

Xu, L., and Choueiry, B. Y. 2003. A new efficient algorithm for solving the simple temporal problem. In *Proceedings of the 10th International Symposium on Temporal Representation and Reasoning and the 4th International Conference on Temporal Logic*.

Yang, X.; Zhou, Z.; Wallace, S.; Lan, Z.; Tang, W.; Coghlan, S.; and Papka, M. E. 2013. Integrating dynamic pricing of electricity into energy aware scheduling for hpc systems. In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, 60. ACM.

Yorke-Smith, N.; Venable, K. B.; and Rossi, F. 2003. Temporal reasoning with preferences and uncertainty. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI)*.

Zhao, Z.; Lee, W. C.; Shin, Y.; and Song, K.-B. 2013. An optimal power scheduling method for demand response in home energy management system. *IEEE Transactions on Smart Grid* 4(3).