Energy-Aware Optimisation of Business Processes

Beatriz Lopez[1]⋆, Aditya Ghose[2], Tony Savarimuthu[3], Mariusz Nowostawski[3], Michael Winikoff[3], and Stephen Cranefield[3]

[1] University of Girona, Girona, Spain
beatriz.lopez@udg.edu
[2] University of Wollongong, Wollongong, Australia
aditya.ghose@gmail.com
[3] University of Otago, Dunedin, New Zealand
first.lastname@otago.ac.nz

Abstract. Due to changes in energy supply, and regulatory mechanism related to energy provisioning, organizations will need to tackle energy management issues. One way of doing so is to allocate resources to business processes taking into account energy costs. However, energy costs are time-dependent, and the resource optimization problem needs to be redesigned. In this paper we formalize the energy-aware resource allocation problem, including time-dependent variable costs; describe how an auction mechanism can be used to allocate resources in a way that optimizes costs; and present a case study.

1 Introduction

The improvement of energy management by organizations, so that costs and CO_2 emissions can be reduced, is a challenge, but has a range of benefits. Reducing CO_2 emissions has not just a social impact on organizations, but has also been demonstrated to improve their stock market performance [2]. Furthermore, organizations that follow certain processes for managing and reducing energy use (e.g. ISO 50001) may also reap other advantages (e.g. Germany has announced tax advantages for ISO 50001 compliant organizations from 2013).

Increasingly, energy prices are changing from flat rates to time-dependent tariffs, which presents companies with the problem of smoothing and shifting peaks from expensive to cheaper hours. Dealing with time-dependent energy costs has been mainly studied regarding household management [10], but few studies have focussed on business process management. An exception is the proposal of Ghose et al. [12] where resources are annotated with CO_2 consumption details which are known to the process manager which aggregates the energy costs.

One way of improving energy management in organizations involves the optimization of resource allocation in business process, not just by responding to price signals to reduce peaks, but also as a potential way to significantly reduce total energy usage.

This paper considers allocation and optimization of resources in business process while taken into account energy costs. The key contribution of the work is the problem formalization, which includes the optimization of resources taking into account the

⋆ Beatriz and Aditya worked on this paper while on leave period at the University of Otago.

time-dependent energy costs. We also provide an auction mechanism for agent-based allocation of resources in business processes, and illustrate with a case study the possible outcomes of the auction.

1.1 Related Work

Research on green business process management is described in [17], where the authors identify the required changes for business processes to be environmentally-aware. Changes are guided by the consideration of key environmental indicators in addition to the traditional key performance ones. The authors propose a service oriented architecture to implement this environmentally-aware approach of business process. Focussing on energy, [1] proposes a framework which includes a control layer in which the energy consumption is optimized according to execution times. In subsequent work [3] describes a tool for validating the desired energy consumption. This tool can be used by designers of Business Process Models (BPM). Moreover, when a deviation is detected, alternative resources can be suggested under varying Quality of Services (QoS) outcomes. All of these approaches are based on web services and a task is assigned to at most one service. By contrast, our work allocates a given task to bundles of services (i.e. resources).

Applying agents to business processes has a history going back over a decade [14]. Most of the multi-agent-based approaches tackle the problem of task sequencing, as do composite services [11], instead of resource allocation. In [8], the difficulties involved in resource allocation for business problems (workflows) are highlighted and the authors indicate that only an agent-based paradigm can appropriately address it. The main issue is uncertainty coming from exclusive-or branching in the business process, that cannot be solved when determining the allocation and makes the problem more complex than resource allocation in supply chain management, to which agents have been successfully applied [4].

The timeline-based scheduling work by Chien et al. [5] is similar to our work in that they have a collection of business process instances, each of which has resource requirements. A key difference between their work and our work lies in the fact that they are able to drop business process instances. If a particular (lower priority) experiment is not able to be scheduled by the algorithm, then it is simply dropped. In our model, we cannot drop business process instances, which means that we have to find a schedule that allows all the given tasks to be carried out. Their work also differs from our work in that their resource model needs to take into considerations more complex physical constraints, e.g. being able to deal with physical equipment that heats up when it is used, and that must not be allowed to overheat. Furthermore, we deal with time-dependent cost of resources, which they do not. The work of Sensoy et al. [7] deals with the allocation of resources to processes. Specifically, they define a flexible framework, using ontologies, that allows for a range of additional constraints or policies to be specified. However, they do not model the cost of resources and do not deal with time-variable costs.

Time-variable costs functions have been recently addressed by the constraint community [21], but the solution proposed can only be applied in centralized environments and is not applicable to our business process model. More generally, traditional job shop literature and workflow resourcing considers a one-to-one mapping of tasks to machines, whereas we allow a task to require multiple resources. Other differences include our allowance for uncertainty in the workflow (through the presence of XOR nodes), and our use of abstract tasks.

2 Problem Definition

We are dealing with energy optimization issues in the context of resource allocation for business processes. First we discuss the available choices; next we provide the formalization of the problem.

An organisation has a number of defined business processes that it performs. For our purposes, we are interested in the collection of instantiated business processes. That is, instead of the templates that describe the general structure of a process, we are interested in the set of actual instances of processes that need to be done, i.e. the workload. More precisely, a business process instance is a directed acyclic graph of tasks. Each task can begin execution once all of its predecessor tasks have completed[4]. Each task takes a certain amount of time to execute, and it requires certain (reusable) resources to be exclusively allocated to this task during the time it executes. Figure 1 shows a single business process in a domain involving servicing of machines. The processes begins with an initial diagnostic assessment ("Preliminaries"). Then the process either outsources the maintenance to an external service provider ("Service provider"), or it allocates an internal technician ("In-technician"), followed by either a sequential or concurrent maintenance process.

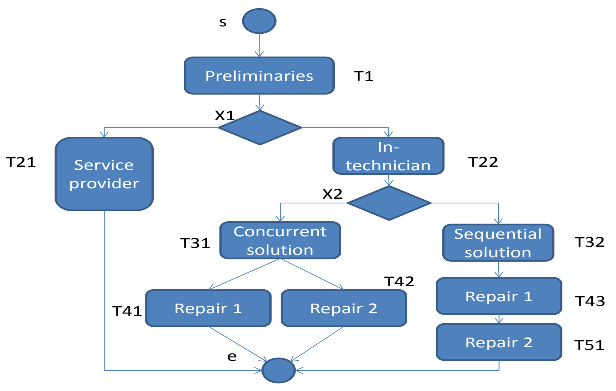


Fig. 1. Business process example

In a business process there are two types of choices that we need to consider. The first is the familiar one where there business process has a decision point, e.g. using an internal technician or outsourcing. The decision is determined by the context of the process, e.g. some machines may need to be outsourced. In other words, when planning and allocating resources, we cannot predict or control these choices (they follow "don't know" nondeterminacy), and so we need to cater for any of the possible choices. Formally, we model these choices using exclusive-or-nodes (XOR-nodes) in the graph (see below).

The second type of choice is where there may be multiple ways of achieving a given abstract task, but the choice is open (i.e. "don't care" nondeterminacy). This sort

[4] But in some situations it may make sense to introduce a delay before starting a task.

of choice provides options for optimisation: given a certain workload, it may be that selecting one particular way of realising a task may result in a reduction in the resources required. For the purposes of this paper we model this type of choice using a single abstract task that has a number of different alternative resource requirements.

We now turn to formalising the problem. Formally we define a set of all the task instances involved in a given workload $T = \{t_1 \dots t_m\}$. Each task has an associated duration, but this duration is not fixed for each task, but depends on the resources used: some resources may enable the task to be achieved more quickly than other resources. We then define a business process instance B as being a graph $B = (V, E)$ where the vertices are either tasks, or one of the two distinguished nodes s ("start") or e ("end"), or one of a set of XOR nodes, $X = \{x_1 \dots x_k\}$. Formally $V = T \cup X \cup \{s, e\}$. As usual, E is a set of pairs of vertices (v, v') where $v, v' \in V$. Each XOR node x_i has an associated set of options: $option(x_i) = \{B^1_{x_i} \dots B^{k_i}_{x_i}\}$ where each of the elements of the set, $B^j_{x_i}$ is a graph, as defined above. In other words, we have a top-level graph B which may have some nodes that are themselves place-holders for one of a set of sub-graphs. We require that the tree of graphs is finite — the leaf graphs are those with no XOR nodes ($X = \{\}$).

The interpretation of XOR nodes is that eventually at runtime each x_i is replace by one of its sub-graphs $B' \in option(x_i)$. This replacement is repeated until there are no XOR nodes remaining. Since there is a choice of $B' \in option(x_i)$ for each x_i, this process is non-deterministic. It results in a "decided" graph which has $X = \{\}$, i.e. no XOR nodes.

Note that a graph B is able to represent both a single business process instance, and a collection of business process instances. We assume that each business process instance has a unique starting node (which is easily ensured), and then the number of edges from s gives the number of business process instances represented by a given graph.

We use $v \rightsquigarrow v'$ to denote that there is a path from v to v', defined in the usual way[5], and use $v_1 \rightsquigarrow v_2 \rightsquigarrow v_3$ as shorthand for $v_1 \rightsquigarrow v_2 \wedge v_2 \rightsquigarrow v_3$.

We require the graph (V, E) to be well-formed, which we define to mean that there are no arcs to the start node ($\neg\exists v.(v, s) \in E$); there are no arcs from the end node ($\neg\exists v.(e, v) \in E$); there are no cycles ($\neg\exists v \in V.v \rightsquigarrow v$); and for any node $v \in V$ (apart from the start and end nodes), there is a path from s to v and from v to e ($\forall v \in (V \setminus \{s, e\}).s \rightsquigarrow v \rightsquigarrow e$).

Each task requires *resources* in order to be carried out. We define a set of known resource types $RT = \{r_1 \dots r_n\}$. We use multisets to represent the set of available resources, $AvailRS$, and the resources that are allocated to each task in a process schedule. A multiset over RT (a "resource multiset") is defined by a characteristic function $R : RT \rightarrow \mathbb{N}$ indicating how many copies of each element of RT appear in the multiset[6]. For convenience we will write multisets using standard set notation,

[5] $v \rightsquigarrow v' \equiv (v, v') \in E \vee (\exists v''.(v, v'') \in E \wedge v'' \rightsquigarrow v')$

[6] This definition of multisets does not allow for real-valued quantities of resources to be represented, but this is not a limitation: we could easily extend the notation to use real numbers, or assume that the unit of measurement is sufficiently fine grained that natural numbers are not a limitation, for instance, measuring coal in units of grams.

but with the possibility of elements appearing multiple times. We will also use the abbreviation a^n to represent n copies of an element, e.g. $\{a^2, b\} = \{a, a, b\}$. We use standard definitions of multiset relations and functions [22]. In particular, we define $R_1 \subseteq R_2$ if and only if $\forall r \in RT.R_1(r) \leq R_2(r)$, i.e. there are at least as many elements of each resource type r in R_2 as there are in R_1. The multiset union operator ($\cup$) is defined by taking the element-wise *maximum* number of copies of resources from its arguments, i.e. $R_1 \cup R_2 = g$ where $\forall r \in RT.g(r) = \max(R_1(r), R_2(r))$. For example $\{a, a, b\} \cup \{a, b, b\} = \{a, a, b, b\}$. We also define an accumulating union ($\uplus$) which retains all copies of resources from its two arguments: $R_1 \uplus R_2 = g$ where $\forall r \in RT.g(r) = R_1(r) + R_2(r)$. For example, $\{a, a, b\} \uplus \{a, b, b\} = \{a^3, b^3\}$.

Each resource type has an associated monetary cost, and an associated energy cost. As discussed earlier, a key requirement is that these costs may vary over time. Formally we define the monetary cost $cost(\tau_1, \tau_2, r_i)$ and energy cost $energy(\tau_1, \tau_2, r_i)$ as being functions from a time interval τ_1 to τ_2 ($\tau_1 < \tau_2$, $\tau_1, \tau_2 \in \mathbb{N}$) and a resource type $r_i \in RT$ to a cost (a real number), representing the *total* cost of *one* unit of resource type r_i over the given time interval.

We also define time-dependent monetary and energy *set-up* costs, $setup\_cost(r, t_1, \tau_1, t_2, \tau_2)$ and $setup\_energy(r, t_1, \tau_1, t_2, \tau_2)$, representing additional costs (or cost savings) that apply when a resource of type r is used for a task t_1 starting at time τ_1 and the next task for the resource is t_2 starting at τ_2 (not necessarily immediately after the end of t_1). A resource type r may also have a minimum set-up time that must be allowed between its use for a task t_1 starting at τ_1 and its next task t_2, denoted $setup\_time(r, t_1, \tau_1, t_2)$. Unlike the set-up cost and set-up energy cost, we assume that this does not depend on the starting time of t_2. Note that, because our representation doesn't distinguish between different instances of a given resource types, set-up costs and times can only be non-zero for resource types where there is a single instance available, i.e. $AvailRS(r) = 1$. We use RT^1 to denote those resource types for which exactly one instance is available. Finally, we generalise the setup costs and times to also apply to XOR nodes: $setup\_cost(r, t_i, start_i, x_j, start_j)$ is defined as the *maximal* cost over those tasks in the subgraphs of the XOR node that are initial for r. A task is initial for r if it is possible for it to be the first task in the subgraph to be executed which uses r, i.e. if we only consider tasks that use r, it is an initial task. We define $setup\_cost(r, x_j, start_j, t_i, start_i)$ in an analogous way (in terms of tasks in x_j that are final for r), and extend to define the setup cost between two XOR nodes in terms of a maximum over the final for r tasks in the first XOR node, and the initial for r tasks in the second XOR node.

We link tasks and resources by defining $need(t_i)$ which denotes the resources that task t_i requires. As discussed earlier, $need(t_i)$ is actually a set of alternative requirements. Furthermore, for each alternative resource requirement, we also associate with it the duration of the task, if the resources in question are allocated to it. Formally $need(t_i) = \{(R_1, d_1^i) \ldots (R_{k_i}, d_{k_i}^i)\}$, where each R_i is a multiset and each d_j^i is a natural number denoting the duration of task t_i if the j^{th} resource multiset is used.

A *schedule* S of a business process assigns to each task a starting and ending time (respecting the sequencing constraints), and resources (such that the available resources are not exceeded at any point in time). Formally, a *schedule* S is a set of task records:

$$S = \{s_1 \ldots s_m\} \cup \{s'_1 \ldots s'_k\} \cup \ \{(start_s, end_s, \{\}, s), (start_e, end_e, \{\}, e)\}$$

There are two types of task records: s_i which correspond to tasks, and s'_j which correspond to XOR nodes. Each task t_i has a single corresponding task record s_i which is a tuple $s_i = (start_i, end_i, RS_i, t_i)$, where RS_i is the resources assigned to the task (constraint: $(RS_i, d^i_j) \in need(t_i)$), $start_i$ is the starting time of the task's performance, end_i is the completion time (constraint: $end_i - start_i = d^i_j$), and t_i is the task. The start and end times must satisfy the constraint that $(t_i, t_j) \in E \ \Rightarrow \ end_i \leq start_j$. Each XOR task x_j has a single corresponding node record s'_j which is a tuple $s'_j = (start_j, end_j, S_j, x_j)$ where $start_j$ and end_j are respectively the start and end times, x_j is the XOR node identifier, and S_j is a *set of sub-schedules*, i.e. a schedule for each sub-graph in $option(x_j)$. We require that $start_j$ must be the smallest of the starting times of a schedule in S_j (and similarly end_j must be the largest finishing time of a schedule in S_j). Finally, for convenience we include records for the start and end nodes. These are assumed to have a zero duration (so $start_s = end_s$ and $start_e = end_e$) and no resources involved (so $RS_s = RS_e = \{\}$).

In fact, the above constraints on the schedule need to be extended to also respect setup times: where a (singleton) resource of type r is used by t_i and then by t_j, the schedule must satisfy the stronger constraint $end_i + setup\_time(r, t_i, start_i, t_j) \leq start_j$. Note that t_i and t_j may not be constrained to occur in sequence. We therefore define this additional feasibility constraint using a singleton resource schedule S^r which, given schedule S and singleton resource type r is a list of the task records for those tasks (both s_i and s'_j) that use resource r, sorted by starting time. The additional feasibility condition is then: $\forall r \in RT^1 \forall (t_i, t_j) \in S^r \ . \ end_i + setup\_time(r, t_i, start_i, t_j) \leq start_j$ where $(t_i, t_j) \in S^r$ denotes the selection of adjacent elements in the list, i.e. $S^r = \langle \ldots t_i, t_j, \ldots \rangle$.

Having defined the constraints that ensure that a schedule is feasible with respect to time (including setup times), we now define the feasibility of a schedule with respect to the available resources ($AvailRS$). We begin by defining functions that accumulate the resources used by a schedule S at time τ:

$$res(\tau, (start_i, end_i, RS_i, t_i)) = \begin{cases} RS_i & \text{if } start_i \leq \tau \leq end_i \\ \{\} & \text{otherwise} \end{cases}$$

$$res(\tau, (start_j, end_j, S_j, x_j)) = \bigcup_{s \in S_j} res(\tau, s)$$

$$Res(\tau, S) = \biguplus_{s \in S} res(\tau, s)$$

The first function (res) takes a time τ and a task record s_i or s'_j and returns the resources required for the task at time τ, which will (for s_i) be either RS_i if the task is being performed at time τ, or the empty set; and for x_j is simply the resources required at time τ in the sub-schedule S_j. The second function (Res) takes a time τ and an entire schedule, and collects the resource requirements for a given time τ across all the tasks.

Finally, a schedule S is feasible with respect to the available resources $\mathit{AvailRS}$ iff $\forall\tau$ such that $\mathit{start}_s \leq \tau \leq \mathit{end}_e$. $\mathit{Res}(\tau, S) \subseteq \mathit{AvailRS}$.

Finally, we need to define the *cost* of a given (feasible) schedule. In fact there are three costs that we consider: the monetary cost, the energy cost, and the time. The monetary cost of a schedule is defined as follows[7]:

$$\mathit{Cost}_\$(S) = \sum_{s\in S} \mathit{Cost}_\$(s) + \sum_{r\in RT^1} \sum_{(t_i,t_j)\in S^r} \mathit{setup\_cost}(r, t_i, \mathit{start}_i, t_j, \mathit{start}_j)$$

$$\mathit{Cost}_\$(s_i) = \sum_{r\in RT} RS_i(r) \times \mathit{cost}(\mathit{start}_i, \mathit{end}_i, r)$$

$$\mathit{Cost}_\$((\mathit{start}_j, \mathit{end}_j, S_j, x_j)) = \max_{S\in S_j} \mathit{Cost}_\$(S)$$

In other words, the cost of a schedule is the sum of the costs for each of the task entries s_i and the XOR entries s'_j (first term), and the sum of the setup costs (second term). Note that, because the setup cost is defined over both task and XOR nodes, and S^r includes both types of nodes, the second term includes the setup costs between tasks and XOR nodes. To compute the cost of an individual task record, we compute for each resource type the cost of a single unit of that resource type across the given time interval, and multiply by the number of resource type units $RS_i(r)$ allocated. For XOR records, we take the maximum across the options.

Similarly, we define the energy cost as:

$$\mathit{Cost}_e(S) = \sum_{s\in S} \mathit{Cost}_e(s) + \sum_{r\in RT^1} \sum_{(t_i,t_j)\in S^r} \mathit{setup\_energy}(r, t_i, \mathit{start}_i, t_j, \mathit{start}_j)$$

$$\mathit{Cost}_e(s_i) = \sum_{r\in RT} RS_i(r) \times \mathit{energy}(\mathit{start}_i, \mathit{end}_i, r_j)$$

$$\mathit{Cost}_e((\mathit{start}_j, \mathit{end}_j, S_j, x_j)) = \max_{S\in S_j} \mathit{Cost}_e(S)$$

Finally, the time cost is simply the makespan:

$$\mathit{Cost}_t(S) = \mathit{end}_e - \mathit{start}_s$$

.

Thus, given a workload represented by graph B, the problem we are tackling is to find a feasible schedule S which minimises these three cost functions ($\mathit{Cost}_\$$, Cost_e and Cost_t). This is a multi-criteria optimization problem in which we cannot a priori assign a greater importance to any given criterion, and so we are looking not for a single "optimal" solution, but for a set of Pareto points. Although it is possible to assign (arbitrary) financial costs to energy and time, and hence reduce this to a single criteria optimisation problem, doing this means that an organisation can no longer rationally consider the tradeoff between energy, time and money.

[7] For convenience we overload $\mathit{Cost}_\$$ to operate on both schedule and task record.

3 Process Provisioning Using Auction Techniques

Auctions have been widely used for task and resource allocation among different entities with the particularity that the price of the resource allocation is decided when clearing the market [4]. Particularly, when auctioneers demand tasks to be performed by resource agents, the mechanism is known as reverse auctions. In an auction process, the cost of a task (monetary, energy and duration) is determined as a result of the competition of all of the resources that can deploy it. The protocol followed in such auctions includes the main following steps:

1. The auctioneer sends a request for proposal, in which the tasks to be performed are specified
2. The bidders answer with some offers
3. The auctioneer decides the set of winner bids
4. The auctioneer acknowledges the winner bidders
5. The bidders commit the time to perform the tasks
6. After the tasks have been performed, bidders send the auctioneer a "done" message that enacts the last step
7. The auctioneer pays for the activity performed by the resources.

There are several choice points from the auction design point of view involved in all of these steps.

First, there is the choice of when the allocation of resources to tasks is performed. Resources can be scheduled in advance or following a dynamic approach, interleaving scheduling and execution of tasks. The latter approach improves the specifications of the tasks to be performed, since the uncertainty about which branch of the XOR-node will be followed is cleared before resources are allocated. Due to the fact that we operate on business process instances and during the execution of the business process we may not have sufficient amount of time to perform the scheduling, we are interested in solutions where we conduct scheduling ahead-of-time and booking of resources is done in advance. This approach requires us to consider resource overlapping on tasks that belong to separate exclusive-or paths of the business process graph.

Second, we can perform a single auction with all of the tasks involved in a business process, or we can proceed with a sequential process, auctioning a task at a time. The latter case is useful if we can obtain a more precise picture of the requirements and constraints of a task once the preceding tasks has been allocated, as in [6]. The presence of XOR and parallel (AND) branches, however, introduces uncertainty on which tasks should be allocated first. Thus we follow the first approach: scheduling all of the tasks in a single auction. To make the allocation process more flexible, so as the resource can express different costs at allocation times (and so, different energy cost), each task is demanded in a time window $[est, let]$, where est is the earliest starting time and let is the latest ending time[8]. Time units are hours (according to the energy day-ahead tariffs, that are also hourly based), and no unit fractions are considered. This is a reasonable

[8] In general, time windows are expressed according to four values: earliest starting time, latest starting time, earliest ending time, and latest ending time. We simplify the approach here by defining the minimum and maximum boundaries of the time window.

assumption in the domains we are working on in which business processes involve human activities that are more than one hour long.

Third, given the set of tasks to be performed, bidders can provide bids on bundles of tasks so that the resulting mechanism is a combinatorial auction. Moreover, bundles enables the resource to express that the assignment of two (or more) consecutive tasks can improve cost. For example, to reduce transport cost to move some resource to a given place where the tasks should be performed (inter-tasks costs). We discuss this issue in Section 3.1.

Fourth, there is a single winner determination problem (WDP) to be solved. The solution of the WDP includes both the allocation times (starting times) and resources for each of the tasks of a business process, taking into account the constraints specified in the problem formulation. Moreover, and thanks to the explicit definition of resource bundles alternatives associated to each task ($need(t_i)$), the winner determination algorithm decides upon a single bundle to a task according to the bids provided. From this point of view, the auction model is a combinatorial auction, since the auctioneer needs to get all of the resources of an alternative ($R_j \in need(t_i)$), or none of them. So our auction model is two-fold combinatorial: bidders express bundles of tasks and auctioneers assign tasks to bundles of resources. This auction model is related to combinatorial exchanges [18]; however, in our work, the business process that wants to buy resources assumes the role of the mediator agent that clears the market. Moreover, if we are taking into account precedence relationship among tasks, our model is related to mixed combinatorial auctions too [9], since in this model the auctioneer needs to decide upon sequence of operations that include transformations of goods. However, in our case, we need to deal with the uncertainty of XOR-paths that is not cleared by the WDP but at the run-time.

Steps four to six of the protocol involve resource acceptance and deployment after the allocation has been cleared. We are dealing with a unique auction and no other request, out of this auction, is being managed by any resource agents. So we do not expect any rejection on behalf of agents. Failures during the tasks execution are out of the scope of this paper (see [19] for preventive issues).

The final step, the payment mechanism, we assume to be incentive compatible, so all of the resource agents provide truthful bids.

3.1 Bidding strategy

Resource agents can receive several request for proposals. So they wait a predefined amount of time before submitting any bid.

After that time, a resource can have several tasks $t_i, \ldots, t_j$ to be considered, each with its own time window. Therefore, there is a considerable amount of combinations that the resource agent can generate. However, for those combinations to be feasible, set up times should be considered.

A bid consists of a list of time-dependent offers $(E_{k_1}, \ldots, E_{k_n})$, together with three matrices for (time-dependent) set up times, energy and monetary costs: M_e, M_m, M_t: $< (E_{k_1}, \ldots, E_{k_n}), M_e, M_m, M_t >$. The bid implicitly expresses that the resource is willing to perform any feasible subset of the offers, unlike in the case of combinatorial auctions where the bids express all of them or nothing.

Each offer E_{k_i} provides information about the task t_{k_i}, the possible time at which the agent is proposing to start performing the task τ_{k_i}, and the energy, monetary and time cost of doing the task at that time correspondingly $(CE_{k_i}, CM_{k_i}, CT_{k_i})$. For example, the offer $E_{k_i} = (t_1@1 : (30, 160, 1))$ expresses the willingness to perform task t_1, at time 1, with $CE_{k_i} = 30$, $CM_{k_i} = 160$, and $CT_{k_i} = 1$.

The rows and columns of each set-up matrix are the bid entries, i.e. the entry -32 in M_m is in the first row and third column, and therefore corresponds to the additional financial cost (actually a saving, since it is negative) if both offer 1 and offer 3 are accepted. An entry of 0 indicates that there is no interaction between offers, and an entry of ∞ indicates that the two offers (row and column indeces) are not compatible. For example, given the bid $(t_1@1 : (30, 160, 1), t_2@2 : (57, 320, 2), t_2@3 : (61, 320, 2), t_3@3 : (30, 160, 1))$ with the following matrices, indicates that the resource can sequence task t_2 at $\tau = 3$ (offer 3) along with t_1 at $\tau = 1$ (offer 1); however, the resource requires an additional hour, and 27 units of energy. On the other hand, the resource expresses its interest on deploying both tasks by offering monetary advantages. Analogously, with t_3 at $\tau = 3$ (offer 4) after processing t_1 at $\tau = 1$ (offer 1), but in this case, the monetary advantage is diminished (16 euros discount instead of 32).

$$M_e = \begin{bmatrix} . & 0 & 27 & 27 \\ 0 & . & 0 & 0 \\ 27 & 0 & . & 0 \\ 27 & 0 & 0 & . \end{bmatrix} \quad M_m = \begin{bmatrix} . & 0 & -32 & -16 \\ 0 & . & 0 & 0 \\ -32 & 0 & . & 0 \\ -16 & 0 & 0 & . \end{bmatrix} \quad M_t = \begin{bmatrix} . & \infty & 1 & 1 \\ \infty & . & 0 & 0 \\ 1 & 0 & . & 0 \\ 1 & 0 & 0 & . \end{bmatrix}$$

3.2 Winner determination problem

Once the auctioneer receives bids from all the resources agents, let's say $b^1, \ldots, b^n$, then it unfolds all the entries E^i_j, to start searching for the best combination of bids. As explained earlier, there are three competing optimization problems that need to be traded off against each other. For example, the goal of optimizing energy is formalised as follows (with analogous definitions for cost and time)

$$arg\ min \sum_{i,j} CE^i_j * x_j \ + \ \sum_{j,l} M_e(E^i_j, E^i_l) * x_j * x_l$$

subject to the constraints formulated in Section 2, and where $x_i \in [0, 1]$ are binary variables to indicate whenever E_i is part of the solution. Observe that a task can be assigned more than once if it requires a bundle of resources. The first term of the expression depicts the cost of performing the tasks without XOR alternatives; the sequence of tasks and XORs should be minimized following the costs functions defined for schedules in Section 2. The second term takes care of the set up cost of assigning to the same resource two tasks with precedence dependencies.

The complexity of the winner determination problems is known to be NP-complete [15], and can be solved by heuristic approaches. Observe that only the resource agents deal with the variability of energy costs, therefore we can use off the shelf constraint solving methods.

4 Case Study

In this section we provide an agent-based solution for scheduling resources to the instances of the business process given in Figure 1. There are two XOR nodes X1 and X2. X1 decides whether the branch containing T21 or the branch containing T22 will be executed. X2 decides whether the branch containing T31 or the branch containing T32 will be executed. Note that tasks T41 and T42 can be executed concurrently (i.e. AND split and join).

For each task, the bundle of resources (or single resources) that are capable of performing the task and the expected duration to complete the task (from the auctioneer's perspective) is given in Table 1 (left). For example, task T21 can be performed by two different bundles of resources, {R2,R1} or {R2,R3}. The resource costs (energy costs and monetary costs) are provided in Table 1 (right). Note that the expected duration of tasks from an auctioneer's perspective may or may not correspond to the actual ability of the resource agents (i.e. the resource agents may provide a different duration as a part of their bids and this information is internal to a resource agent). For example, the common information indicates that T21 takes takes 5 hours regardless of which resource bundle is taken, but the resources themselves indicate that the task actually only takes 4 hours.

We assume day-ahead hourly prices (i.e. energy cost in kWh per hour) is available similar to the work of Gottwalt et al. [10] (Figure 2). Additionally, we also assume that the resource allocation process starts at 8am.

Task	(Resources, Duration)
T1	({R1}, 1)
T21	({R2,R1}, 5), {R2,R3}, 5)
T22	({R1},1),({R4},1)
T31	({R1},1),({R4},1)
T32	({R1},1),({R4},1)
T41	({R3},2),({R4},2)
T42	({R3},3),({R4},3)
T43	({R1},2),({R4},2)
T51	({R3},1),({R4},1)

Resource	Energy	Money	Duration
R1	1	160	(T1,1),(T21,4),(T22,3),(T31,2) (T32,1),(T43,1)
R2	10	200	(T21,4)
R3	5	90	(T21,4),(T41,3),(T42,4),(T51,1)
R4	3	50	(T22,1),(T31,1),(T32,2),(T41,3), (T42,3),(T43,4),(T51,2)

Table 1. Resources required by each task (left) and resources' features (right). Energy is expressed in kWh; money in euros; duration in hours

We illustrate the solution following the auction mechanism. All the tasks from all of the BP instances are allocated in a single auction. For the business process given in Figure 1, there are 9 tasks for which resources need to be allocated. They are T1, T21, T22,

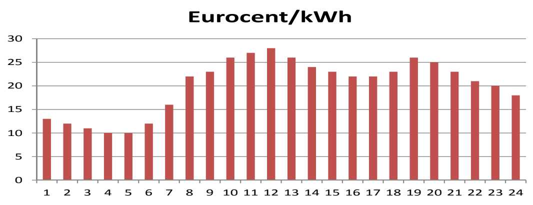


Fig. 2. Day ahead hourly tariffs.

T31, T32, T41, T42, T43, T51. For
each task, a time window is generated according to the duration estimated by the auctioneer (see Table 1 (left)) with a slack time of 2 hours. For example, since the schedule begins at 8am and T1 takes 1 hour, the request (T1, [8,11]) is sent to R1.

It should be noted that each agents may receive several requests. Agents process the requests and generate a single bid, which includes the set up time matrices as explained in Section 3.1. We have considered set-up costs in R4 (between T32 and T43, and between T43 and T51) and R3 (between T43 and T51). In all of the cases, the set-up cost consists on an extra monetary, energy and time unit.

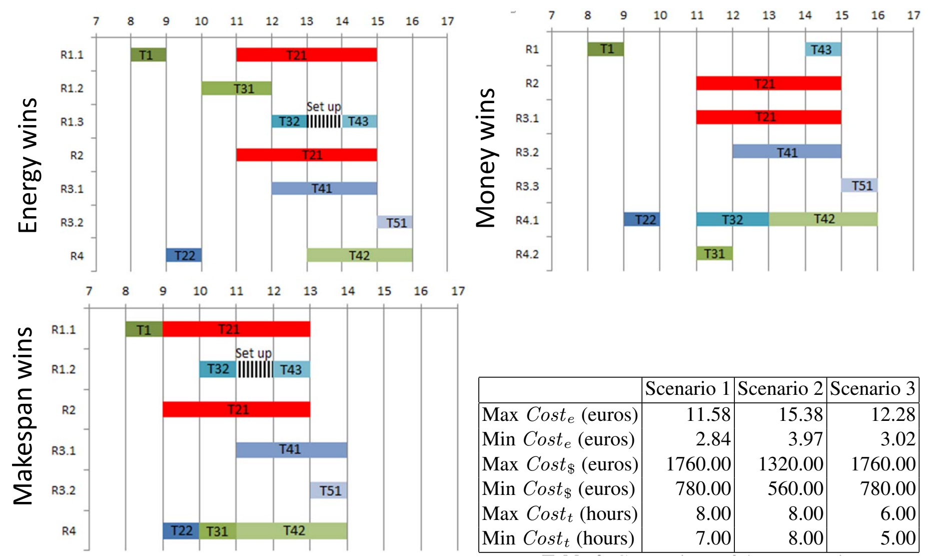
Once the auctioneer collects all of the bids, the winner is determined. For determining the winner of the auction we consider three different strategies. These three strategies consider three attributes, the energy cost, the monetary cost and the makespan. However, the relative importance of these attributes differs in each of the strategies.
Energy wins: the bid with the cheapest energy cost is selected. In case of a tie between energy costs, the one with the cheapest monetary cost will be preferred. Again if there is a tie, the makespan will be considered.
Money wins: the bid with the cheapest monetary cost is selected. In case of ties (same monetary costs), energy costs will be considered for comparison. If there is a tie in energy costs, makespan will be considered.
Makespan wins: the bid with the shortest makespan is selected. In case of a tie, the cheapest energy cost is considered. If there is a tie between energy costs, the cheapest monetary cost is considered. We consider the makespan whose starting time is closest to the earliest starting time of the activities scheduled for a given day (i.e. 8am in our case).

We note six different strategies can be considered for a combination of three attributes. In this paper we compare three strategies and we believe these strategies are sufficient to show the consequences of using energy costs in resource allocation in business processes.

The resultant resource allocations at the end of the auction process for each of the strategies are shown in Figure 3. Note that a resource agent can have overlapping bookings, because of the XOR branches. The overbooking of an agent is represented as different horizontal bars in Figure 3. For example, R1 has three bookings (R1.1, R1.2 and R1.3). Note that the booking R1.1 overlaps with the other two. However, these bookings are for the tasks in XOR nodes, so these would not result in a conflicting situation at run-time.

If we look at T21, we see how the bundles are allocated differently in each of the strategies. In the energy strategy T21 is assigned to R1 and R2 (at time 11), in the monetary strategy to R2 and R3 (at time 11), and in the makespan strategy to R1 and R3 again, but at a different time (at time 9). In the Figure, is it possible to observe how parallel branches (tasks T41 and T42) do not need to be executed necessarily at the same time. Finally, note that set-up costs have only been considered in the energy and makespan strategies, when sequencing tasks T32 and T43 in resource R1.

To quantify the results obtained, we have measured the worst (max) and best (min) cost on each strategy. We have a range of possible costs for each strategy because of the uncertainty associated with XOR nodes: we don't know in advance which branch will



	Scenario 1	Scenario 2	Scenario 3
Max $Cost_e$ (euros)	11.58	15.38	12.28
Min $Cost_e$ (euros)	2.84	3.97	3.02
Max $Cost_\$$ (euros)	1760.00	1320.00	1760.00
Min $Cost_\$$ (euros)	780.00	560.00	780.00
Max $Cost_t$ (hours)	8.00	8.00	6.00
Min $Cost_t$ (hours)	7.00	8.00	5.00

Fig. 3. Resource assignment. Top left: Energy wins strategy. Top right: Money wins strategy. Bottom left: Makespan wins strategy. Bottom right: comparison of strategies.

be taken. The results obtained are shown in Figure 3 (bottom right). If we compare the worst case energy costs between strategies 1 and 2, the energy cost in strategy 2 is 33% more than strategy 1 (11.58 vs. 15.38). The monetary cost in strategy 1 is 33.3% more than strategy 2 (1760 vs. 1320). In the best case monetary cost strategy, the difference in monetary cost between strategies 1 and 2 increases to 39% (780 vs. 560). However, in this case strategy 1 has a lower makespan than strategy 2. Money and energy are moving in different scales (hundreds versus tens). However, with our experiments here we are showing how the resource selection changes the energy costs. In other strategies, energy could be more expensive.

In the makespan strategy (strategy 3), the monetary cost is the same as strategy 1, however, the energy cost increases by 6% both in the best and worst case strategies when compared to strategy 1. It should be noted that resources are more idle in strategy 1 than strategy 2 (worst case make span of 8 vs. 6).

It should be noted that the results presented in Figure 3 (bottom right) show the three what-if scenarios modelled which can be valuable for the decision makers (e.g. managers of the business processes). They can use this information to weigh trade-offs between alternatives. For example, a manager currently employing the makespan strategy (strategy 3), can now weigh the trade-off in moving towards a strategy where energy costs are minimized (i.e. strategy 1). In this case, choosing strategy 1 would result in longer makespan with no extra monetary cost, which the manager may consider as a viable alternative since overall energy consumed has been minimized (i.e. adheres to the green 'energy reduction' norm). Our main contribution in this paper is the consideration of the energy cost as a key attribute in decision making and the optimization of resource allocation in business process instances.

5 Discussion & Future Work

This paper presents an approach that considers energy as one of the attributes for optimizing resource allocations and scheduling prior to business process execution. Towards that end, this paper presents the problem formalization and discusses an auction mechanisms that can be employed. One of the key contributions of this paper is the consideration of time-dependent energy costs as a part of scheduling and resource allocation, which has not been previously considered by work on BPM.

A one-shot auction mechanism is used in this paper. An auctioneer auctions all the tasks of all business process instances for a day (workload) in one shot. Other flexible models can be considered in the future. Examples of these models include a) on-line auctions to handle business processes that arrive at any time, b) sequential auctions that deal with business processes that arrive one after the other, and c) concurrent auctions where there is an auction for each of the XOR paths of a business process. Investigating the consequences of employing these different auctions and comparing their efficiency in achieving global optimization (e.g. facilitating overall energy reduction) will be of interest in the future.

The limitations of the auction mechanism proposed in the paper include the following. First, more detailed experiments should be conducted on various what-if scenarios in order to understand the implications of scenarios considered (e.g. running several concurrent instances of various business processes). Second, using bundles of resources for a particular task should be investigated in detail (currently only one of the tasks in the case study uses a bundle of resources). Third, incorporating negotiation techniques between different entities can be considered. For example, inter-agent negotiation for swapping tasks between agents and inter-auctioneer negotiation (in the case of concurrent auctions) for optimized resource allocation can be considered in the future. Fourth, the impact of set-up costs between consecutive tasks will need to be investigated in detail.

The domain of considering energy as a key component in scheduling and resourcing business process executions offers interesting challenges. For example, for an organization to adhere to the ISO norms of being energy efficient and also consuming green energy [13], an organization may choose to negotiate a deal with the energy provider based on the energy signature (or energy consumption shape curve) for a particular day. Energy providers can offer special rates for those companies that adhere to their expected energy shape (i.e. energy usage at different hours of the day). This leads to other interesting scenarios such as companies offering auctions on excess surplus energy to those that need some additional energy, similar to the dynamic coalition formation scenario considered for the construction of virtual power plant [16]. It should be noted that the use of the energy signature at the organization level considers the energy consumption of the business process and the resources. Note that energy costs (day ahead, hourly tariffs) can be different depending on the locality of the resource, while the use of renewal energy generators can provide some advantages to the resources by offering low energy costs. Furthermore, the energy consumed by external resources (i.e. resources from other organizations) will account differently to the optimization problem since the energy shape it adheres to will be different from the energy shape of the organization that owns the business process instance. In other words, the external resource may have a different energy objective function than the internal resource. One approach to address this problem is to enable Workflow Management Systems in charge of business process resource allocations to coordinate activities with Energy Management Systems (EnMS) that are in-charge of the company energy policy [20]. EnMS can facilitate choosing external resources that closely align with energy objective functions of a given organization.

References

1. Ardagna, D., Cappiello, C., Lovera, M., Pernici, B., Tanelli, M.: Active energy-aware management of business-process based applications. In: ServiceWave '08. pp. 183–195. Springer-Verlag (2008), http://dx.doi.org/10.1007/978-3-540-89897-9_16
2. Bose, I., Pal, R.: Do green supply chain management initiatives impact stock prices of firms? Decision Support Systems 52(3), 624 – 634 (2012), http://www.sciencedirect.com/science/article/pii/S0167923611002004
3. Cappiello, C., Fugini, M., Gangadharan, G., Ferreira, A., Pernici, B., Plebani, P.: First-step toward energy-aware adaptive business processes. In: On the Move to Meaningful Internet Systems: OTM 2010 Workshops, LNCS, vol. 6428, pp. 6–7. Springer (2010), http://dx.doi.org/10.1007/978-3-642-16961-8-4
4. Chevaleyre, Y., Dunne, P.E., Endriss, U., Lang, J., Lemaître, M., Maudet, N., Padget, J., Phelps, S., Rodríguez-Aguilar, J.A., Sousa, P.: Issues in multiagent resource allocation. Informatica 30, 2006 (2006)
5. Chien, S.A., Tran, D., Rabideau, G., Schaffer, S.R., Mandl, D., Frye, S.: Timeline-based space operations scheduling with external constraints. In: ICAPS. pp. 34–41. AAAI (2010)
6. Collins, J., Gini, M.: Scheduling tasks using combinatorial auctions: the magnet approach. In: In Gedas Adomavicius and Alok Gupta, editors, Handbooks in Information Systems Series: Business Computing, Elsevier (2008)
7. Şensoy, M., Vasconcelos, W.W., Norman, T.J., Sycara, K.: Reasoning support for flexible task resourcing. Expert Syst. Appl. 39(2), 1998–2010 (Feb 2012), http://dx.doi.org/10.1016/j.eswa.2011.08.041
8. Delias, P., Doulamis, A., Matsatsinis, N.: What agents can do in workflow management systems. Artif. Intell. Rev. 35, 155–189 (February 2011), http://dx.doi.org/10.1007/s10462-010-9189-3
9. Giovannucci, A., Cerquides, J., Endriss, U., Rodríguez-Aguilar, J.A.: A graphical formalism for mixed multi-unit combinatorial auctions. Autonomous Agents and Multi-Agent Systems 20, 342–368 (May 2010), http://dx.doi.org/10.1007/s10458-009-9085-x
10. Gottwalt, S., Ketter, W., Block, C., Collins, J., Weinhardt, C.: Demand side management - A simulation of household behavior under variable prices. Energy Policy 39(12), 8163 – 8174 (2011), http://www.sciencedirect.com/science/article/pii/S0301421511008007
11. Guo, L., Robertson, D., Chen-Burger, Y.H.: Using multi-agent platform for pure decentralised business workflows. Web Intelli. and Agent Sys. 6, 295–311 (August 2008), http://dx.doi.org/10.3233/WIA-2008-0142
12. Hoesch-Klohe, K., Ghose, A., Lê, L.S.: Towards green business process management. In: Proceedings of the IEEE International Conference on Services Computing (SCC). pp. 386 – 393 (2010)
13. ISO 50001:2011: Energy management systems – requirements with guidance for use. http://www.iso.org/iso/ [Accessed: 21.2.2012]
14. Jennings, N.R., Faratin, P., Norman, T.J., O'Brien, P., Odgers, B.: Autonomous agents for business process management. Int. Journal of Applied Artificial Intelligence 14(2), 145–189 (2000), http://eprints.ecs.soton.ac.uk/3739/

15. Lehmann, D., Mueller, R., Sandholm, T.: The winner determination problem. In: Cramton, Shoham, Steinberg (eds.) Combinatorial Auctions, chap. 12. MIT Press (2006)
16. Mihailescu, R.C., Vasiran, M., Ossowski, S.: Dynamic coalition formation and adaptation for virtual power stations in smart grids. In: ATES Workshop (May 2011)
17. Nowak, A., Leymann, F., Schumm, D.: The differences and commonalities between green and conventional business process management. In: Dependable, Autonomic and Secure Computing (DASC), 2011 IEEE Ninth International Conference on. pp. 569 –576 (dec 2011)
18. Parkes, D.C., Kalagnanam, J., Eso, M.: Achieving budget-balance with vickrey-based payment schemes in exchanges. In: IJCAI- Volume 2. pp. 1161–1168 (2001), http://dl.acm.org/citation.cfm?id=1642194.1642250
19. Ramchurn, S.D., Mezzetti, C., Giovannucci, A., Rodriguez, J.A., Dash, R.K., Jennings, N.R.: Trust-based mechanisms for robust and efficient task allocation in the presence of execution uncertainty. Journal of Artificial Intelligence Research 35, 119–159 (June 2009), http://eprints.ecs.soton.ac.uk/17288/
20. Roche, R., Blunier, B., Miraoui, A., Hilaire, V., Koukam, A.: Multi-agent systems for grid energy management: A short review. In: IECON. pp. 3341 –3346 (nov 2010)
21. Simonis, H., Hadzic, T.: An energy cost aware cumulative. In: Third Int. Workshop on Bin Packing and Placement Constraints (BPPC) (2010), 6 pages
22. Syropoulos, A.: Mathematics of multisets. In: Calude, C., Paun, G., Rozenberg, G., Salomaa, A. (eds.) Multiset Processing: Mathematical,Computer Science, and Molecular Computing Points of View. LNCS, vol. 2235, pp. 347–358. Springer (2001)