Nonserial Dynamic Programming with Applications in Smart Home Appliances Scheduling – Part II: Nonserial Dynamic Programming

Kin Cheong Sou, Henrik Sandberg and Karl Henrik Johansson

***Abstract*— In this paper a dynamic programming (DP) solution approach to a nonconvex resource allocation problem (RAP) is presented. The problem in this paper generalizes the smart home appliances scheduling problem introduced in the companion paper (i.e., Part I). The computation difficulty with solving the RAP depends on the decision variable coupling, which can be described by an interaction graph. This paper proposes a DP algorithm to solve the RAP in the special setting where the interaction graph is a tree. This extends the applicability result of DP to RAP beyond the standard serial case where the interaction graph is a line. The extension of the result is achieved by establishing that even in the tree case DP computation effort is polynomial in the number of decision variables. For RAP in its general form, this paper proposes a modification to the nonserial DP procedure originally introduced by Bertele and Brioschi. Contrary to that of the previous method, the computation effort of the proposed method can be characterized by a well-known minimum feedback vertex set problem. Case studies on known examples indicate that both nonserial DP methods are similarly efficient.**

I. Introduction

A. Problem Statement

This paper is concerned with the dynamic programming (DP) solution procedure to the following nonconvex resource allocation problem:

$$\begin{aligned} \underset{x_1,x_2,\ldots,x_n}{\text{minimize}} \quad & \sum_{k=1}^{m} F_k(X_k) \\ \text{subject to} \quad & X_k \subset \{x_1,x_2,\ldots,x_n\}, \quad \forall\, k, \\ & \sum_{j=1}^{n} g_j(x_j) \le b, \\ & x_j \in C, \quad j=1,2,\ldots,n, \end{aligned} \tag{1}$$

where X_k are subsets of the set of all decision variables and $F_k(X_k)$ are arbitrary given functions with arguments being the members of X_k. For instance, if $X_k = \{x_1, x_3, x_5\}$, then $F_k(X_k) := F_k(x_1, x_3, x_5)$. In (1), C is a given finite discrete set. While it is possible to generalize to the case where there are different C_j for different x_j, this paper restricts the description to the case with the single set C in order to avoid notation complication. The coupling of the sets X_k's can be described by an interaction graph H. H is an undirected graph, and it has n nodes indexed by $1,2,\ldots,n$, with node i corresponding to decision variable x_i. Two nodes i and j are connected by an (undirected) edge if and only if there exists an index k such that both x_i and x_j are members of X_k. See Figure 1 for an illustration of the interaction graph. In (1), g_j are arbitrary given functions in the resource budget constraint. b is a given scalar. It is assumed that there exists a discrete set B whose elements are nonnegative integer multiples of some $\Delta b > 0$ such that $b \in B$ and $g_j(x_j) \in B$ for all $x_j \in C$ and all $j = 1,2,\ldots,n$.

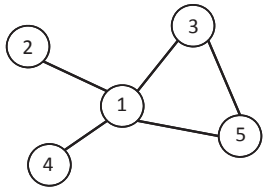

Fig. 1. The interaction graph of an instance of (1) with five decision variables. The objective function is $F_1(x_1,x_2) + F_2(x_1,x_4) + F_3(x_1,x_3,x_5)$. Each component of the objective function defines a clique in the interaction graph.

B. Motivations

The resource allocation problem in (1), a resource-constrained version of the problem studied in [1], was considered in [2]. Problem (1) is also motivated from the following smart home appliances scheduling problem described in the companion paper [3], as the later problem can be reduced into problem (1) (see Appendix A for more detail):

$$\begin{aligned} \underset{t_1,t_2,\ldots,t_n}{\text{minimize}} \quad & \sum_{i=1}^{n} f_i(t_i) \\ \text{subject to} \quad & l_{ij} \le t_j - t_i \le u_{ij}, \quad \forall\, \{i,j\} \\ & t_i \in C, \quad \forall\, i, \\ & \sum_{i=1}^{n} g_i(t_i) \le b, \end{aligned} \tag{2}$$

where f_i are arbitrary given functions, l_{ij} and u_{ij} are given constants (l_{ij} can be $-\infty$ and u_{ij} can be $+\infty$ to indicate that the corresponding constraints are in fact absent), C is a given discrete set, and the resource budget constraint is the same as the one in (1), except that the symbols for the decision variables have changed. Problem (2) is one of the smart home appliances scheduling problems which have received much attention recently (e.g., [4]–[10]). Appliances scheduling forms an integral part of the automatic decision support system in smart grid applications. Appliances scheduling allows household consumers to consciously control their energy consumption patterns without being overwhelmed by the amount of information provided by smart grid technologies

Kin Cheong Sou is with the Department of Mathematical Sciences, Chalmers University of Technology and the University of Gothenburg, Sweden. The other authors are with the ACCESS Linnaeus Center and the Automatic Control Lab, the School of Electrical Engineering, KTH Royal Institute of Technology, Sweden. kincheong.sou@chalmers.se, {hsan,kallej}@kth.se

(e.g., the demand response signals from smart meters). This in turn enables the power network operators to indirectly control the peak energy demand, a task particularly relevant in situations with high level of renewable energy penetration. Thus, an efficient algorithm for (1) (and hence for (2)) is a critical component for the realizability of smart grid.

C. Contributions of the Paper

Problem (1) is NP-hard in general [2]. In [2], the standard DP procedure was used to solve problem (1) in the case where X_k's are separable. That is, $m = n$, $X_k = \{x_k\}$ for all k and the corresponding interaction graph consists of isolated nodes. The computation effort for this specialization is $O(n|B||C|)$. It is also known that in a more general case where X_k are serial with $X_k = \{x_k, x_{k+1}\}$ for $k = 1, \ldots, n-1$ (i.e., the corresponding interaction graph is a single line), the standard DP can still be utilized with computation effort being $O(n|B||C|^2)$ [1]. In this paper, it is demonstrated that a generalization of the serial case still admits DP solution with similar computation effort of $O((n-1)|B|^2|C|^2)$. This is the case where the interaction graph is a tree (i.e., a connected graph without cycles). The details will be presented in Section II.

In addition to generalizing the result on the applicability of DP, the DP solution procedure in Section II is also needed for the proposed nonserial DP procedure for the general version of (1) with no restriction on the interaction graph. The proposed nonserial DP procedure will be described in Section III. This procedure, a modification of the original nonserial DP method in [1], organizes the solution finding of (1) into a combination of enumeration and DP with tree interaction graph which is the method described in Section II. Even though in the worst case the computation effort required is exponential in problem size because of the enumeration, the computation effort of the proposed approach in Section III can be characterized by solving the well-studied minimum feedback vertex set problem on the interaction graph. This characterization is much more straightforward than the one required by the original nonserial DP in [1]. Case studies on example instances indicate that the presented approach has the same computation effort requirement as the one in [1].

II. Dynamic Programming with Tree Interaction Graph

In this section the interaction graph H of (1) is assumed to be a tree. The solution procedure to be described can in fact handle the case where H is a forest (i.e., an acyclic graph with more than one connected components). If H is a forest, the separate connected components can be connected by artificial edges by adding constant objective function components to the original objective function. For example, suppose node 1 and node 2 are on separate connected components, the added objective function component can be $F_{1,2}(x_1, x_2) \equiv 0$.

The tree structure of the interaction graph H of (1) implies that in the most compact description it holds that (a) the number of objective function components F_k's is $n-1$ (i.e., $m = n-1$), and (b) each distinct F_k is associated with a distinct pair of nodes $\{i_k, j_k\}$ corresponding to an edge in H. Therefore, each function $F_k(X_k)$ can be specialized to $F_{i_k,j_k}(x_{i_k}, x_{j_k})$, where i_k and j_k are the two end nodes of the edge associated with X_k. The argument for the claims can be found in Appendix B. The proposed DP approach to solve (1) with tree interaction graph is described next, and after that Figure 2 shows an illustration of the proposed DP approach.

step 0 (initialization):

- Set $H' := H$.
- Set u to be the index of a leaf node in H' (i.e., a node with only one neighbor).
- For all $k = 1, 2, \ldots, n$, set successor counter $s_k = 0$. Set successor stack $S_k = [\,]$.
- For all $(x_k, b_k) \in C \times B$ with $k = 1, 2, \ldots, n$, set $J_k^0(x_k, b_k)$ according to

$$J_k^0(x_k, b_k) = \begin{cases} 0 & \text{if } g_k(x_k) \le b_k, \\ \infty & \text{if } g_k(x_k) > b_k. \end{cases} \tag{3}$$

- Go to **step 1**.

step 1 (finding predecessor): If the leaf node u does not have any neighbor, then go to **step 3**. Otherwise, set v to be the index of the neighbor of u (there can be only one neighbor since u is a leaf node), and go to **step 2**.

step 2 (absorbing leaf nodes):

2-1: Let $N_l(v)$ denote the index set of all neighbors of v which are leaf nodes of H'.

2-2: For each $w \in N_l(v)$, perform the following:

- For each pair $(x_v, b_v) \in C \times B$, solve the minimization problem

$$\begin{aligned} \underset{x_w \in C,\, b_w \in B}{\text{minimize}} \quad & F_{v,w}(x_v, x_w) + J_w^{s_w}(x_w, b_w) \\ & + J_v^{s_v}(x_v, b_v - b_w), \end{aligned} \tag{4}$$

where s_v and s_w are the most updated successor counters for v and w, respectively. Also, let

$$(x_w^\star, b_w^\star) := (x_w^\star(x_v, b_v), b_w^\star(x_v, b_v)) \text{ denote a minimizing pair of (4).} \tag{5}$$

- Define $J_v^{s_v+1}(x_v, b_v)$ as the optimal objective value of (4). That is,

$$\begin{aligned} J_v^{s_v+1}(x_v, b_v) \quad = \quad & F_{v,w}(x_v, x_w^\star) + J_w^{s_w}(x_w^\star, b_w^\star) \\ & + J_v^{s_v}(x_v, b_v - b_w^\star). \end{aligned} \tag{6}$$

- Increment $s_v \leftarrow s_v + 1$ and update successor stack $S_v(s_v) = w$. That is, w is the latest successor of v (or v is the predecessor of w). Also, update H' by deleting nodes w and edges $\{w, v\}$. The process in **2-2** is referred to as "absorbing" the leaf node $w \in N_l(v)$. Note that after w is absorbed it disappears from H'. Hence, S_w and s_w will not change again. S_w is an ordered list of the successors of w (in the order of absorption), and s_w is the number of members in S_w.

2-3: Find a leaf node u in the updated H' (u can always be found since the updated H' is still a tree). With the updated u and updated H', go to **step 1**.

step 3 (optimal solution backtracking):

- Let v_1 denote the only node in H with no predecessor (the precedence relationship was defined in **step 2-2**). Define a graph $T_{v_1}^{s_{v_1}}$ with the nodes in H. Let (u,v) be a directed edge of $T_{v_1}^{s_{v_1}}$ if and only if u is a predecessor of v. It will be shown that $T_{v_1}^{s_{v_1}}$ is connected.
- Perform a breadth-first search (BFS) on $T_{v_1}^{s_{v_1}}$ starting from v_1 with three additional rules: (a) define $x_{v_1}^{\rm opt}$ as a minimizer of $J_{v_1}^{s_{v_1}}(\cdot,b)$ and $b_{v_1}^{\rm opt}=b$. (b) when a node has more than one unexplored successors, BFS always explores the successor which is absorbed last (i.e., exploration in the reverse order of S_v for any node v). (c) when the k-th successor of a node v (i.e., $S_v(k)$) is explored, define

$$\begin{aligned} x_{S_v(k)}^{\rm opt} &= x_{S_v(k)}^{\star}\Big(x_v{}^{\rm opt}, b_v^{\rm opt} - \sum_{l=k+1}^{s_v} b_{S_v(l)}^{\rm opt}\Big), \\ b_{S_v(k)}^{\rm opt} &= b_{S_v(k)}^{\star}\Big(x_v{}^{\rm opt}, b_v^{\rm opt} - \sum_{l=k+1}^{s_v} b_{S_v(l)}^{\rm opt}\Big). \end{aligned} \tag{7}$$

- After the modified BFS, the optimal solution is returned as $x_k^{\rm opt}$ for all k. This will be certified in Theorem 2.

An illustration of the **step 1/step 2** loop for the DP approach applied to an example problem instance with an 8-node tree interaction graph is shown in Figure 2. The graph $T_{v_1}^{s_{v_1}}$ defined in **step 3** is important in the subsequent analysis and it has the following property:

Lemma 1: The graph $T_{v_1}^{s_{v_1}}$ defined in **step 3** is a directed tree rooted at v_1.

Proof: The fact that $T_{v_1}^{s_{v_1}}$ is a tree when the edge directions are ignored is established because an edge (u,v) in $T_{v_1}^{s_{v_1}}$ corresponds to an edge $\{u,v\}$ in H (and H is a tree). Since every node other than v_1 has a unique predecessor and $T_{v_1}^{s_{v_1}}$ is a tree, for each node u there is a unique directed path from v_1 to u. Therefore, $T_{v_1}^{s_{v_1}}$ is rooted at v_1. ■

Associated with $T_{v_1}^{s_{v_1}}$ the following concept is important in the analysis: For each node v in $T_{v_1}^{s_{v_1}}$ and k such that $0 \le k \le s_v$, let T_v^k, referred to as the (cumulative) partial subtree rooted at v with index k, denote a subgraph of $T_{v_1}^{s_{v_1}}$ with the following properties: The set of nodes of T_v^k, denoted $N(T_v^k)$, includes node v, the first k successors of v (i.e., $S_v(1),\ldots,S_v(k)$) and all descendants of these successors. The set of edges of T_v^k, denoted $E(T_v^k)$, includes the edges of $T_{v_1}^{s_{v_1}}$ whose both ends are in $N(T_v^k)$. The concept T_v^k generalizes that of $T_{v_1}^{s_{v_1}}$ since the latter is indeed the former with $v=v_1$ and $k=s_{v_1}$. By definition, $S_v(0)$ is an empty stack and hence T_v^0 for any node v contains node v itself. Figure 3 illustrates some topological concepts associated with T_v^k with the 8-node example in Figure 2.

Associated with each partial subtree T_v^k, $x_v \in C$ and $b_{T_v^k} \in B$, the subproblem $P(T_v^k, x_v, b_{T_v^k})$ is defined as

$$P(T_v^k,x_v,b_{T_v^k}): \quad \begin{array}{ll} \underset{\substack{x_u\,:\,u\in N(T_v^k)\\ x_u\in C}}{\text{minimize}} & \sum_{\substack{(i,j)\in E(T_v^k),\\ \text{or } (j,i)\in E(T_v^k)}} F_{i,j}(x_i,x_j) \\ \text{subject to} & \sum_{u\in N(T_v^k)} g_u(x_u) \le b_{T_v^k} \\ & x_v \quad \text{given.} \end{array} \tag{8}$$

By convention, if $E(T_v^k)$ is empty (i.e., when T_v^k is a single node), then the objective is a constant zero function. $P(T_{v_1}^{s_{v_1}}, x_{v_1}, b)$ is the original resource allocation problem in (1) except that x_{v_1} is fixed. Therefore, an enumeration over x_{v_1} with $P(T_{v_1}^{s_{v_1}}, x_{v_1}, b)$ leads to a solution to (1), provided that the optimal choices for the rest of the decision

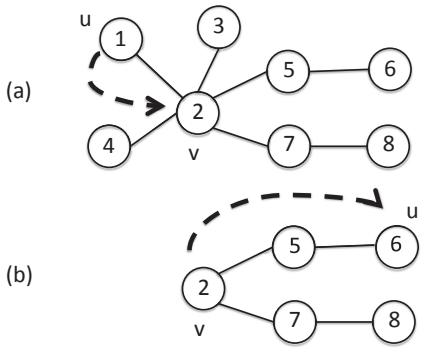

Fig. 2. Illustration of the DP approach applied to an example problem instance with a tree interaction graph with eight nodes. In **step 0** the initialization picks node 1 as the leaf node, designated u in subfigure (a). In **step 1**, node 1 is found to have a neighbor node 2, which is designated v, the predecessor of u. Subfigure (b) shows the result at the end of one pass of **step 2**. All leaf-neighbors of node $v=2$ (i.e., 1, 3 and 4 but not 5 or 7) are "absorbed" into v during **step 2**. The order in which nodes 1, 3 and 4 are absorbed is arbitrary, but this information is recorded in the successor stack S_2 (e.g., $S_2=[1,3,4]$). The intermediate interaction graph H' is updated: nodes 1, 3 and 4, as well as the edges $\{1,2\}$, $\{2,3\}$ and $\{2,4\}$ are deleted in the updated H', as shown in (b). Not shown in (b), the functions $J_2^j(\cdot,\cdot)$ are also defined as in (6), for $j=1,2,3$. Since in the updated H' node 2 is not a leaf, a new leaf node (i.e., node 6) is designated u at the end of **step 2**. The new leaf node always exists, because by construction the updated H' is also a tree, and every tree contains at least one leaf node. The procedure continues to iterate through **step 1** and **step 2**, absorbing the nodes in the order of 6, 5, 2, 7. Note that during this process, node 2 is again designated as v (as the predecessor of node 5) and the corresponding $J_2^4(\cdot,\cdot)$ is defined. Finally only node 8 is left in the latest version of H'. **step 3** is then executed until the DP procedure is complete.

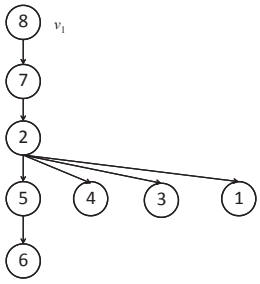

Fig. 3. Illustration of T_8^1, the directed tree rooted at node 8. The successors of node 2 include 5, 4, 3 and 1, ranked according to the reverse order of absorption. The descendants of node 2 include 1, 3, 4, 5 and 6. The partial subtree S_2^3 includes nodes 2, 1, 3, 4 and edges $(2,1)$, $(2,3)$, $(2,4)$.

variables in (8) can be determined. Analogous to standard DP, the minimization of $P(T_{v_1}^{s_{v_1}}, x_{v_1}, b)$ is carried out through a "cost-to-arrive" function, which turns out to be $J_{v_1}^{s_{v_1}}(x_{v_1}, b)$ defined in **step 2**. Indeed, the following statement asserts that $J_v^k(x_v, b_{T_v^k})$ is the optimal objective value of subproblem $P(T_v^k, x_v, b_{T_v^k})$.

Theorem 1: Consider applying the DP algorithm with the loop between **step 1** and **step 2** finishes. Let v be a node in $T_{v_1}^{s_{v_1}}$ and let k be an integer such that $0 \le k \le s_v$ with s_v being the number of successors of v in $T_{v_1}^{s_{v_1}}$. Let T_v^k denote a partial subtree rooted at v. For any given $x_v \in C$, $b_{T_v^k} \in B$, the value of $J_v^k(x_v, b_{T_v^k})$ is the optimal objective value of subproblem $P(T_v^k, x_v, b_{T_v^k})$ defined in (8). In addition, for $k \ge 1$, $w := S_v(k)$, it holds that

$$J_v^k(x_v, b_v) = F_{v,w}(x_v, x_w^\star) + J_w^{s_w}(x_w^\star, b_w^\star) + J_v^{k-1}(x_v, b_{T_v^k} - b_w^\star), \tag{9}$$

where $(x_w^\star, b_w^\star)$ are defined by (5) with x_v and $b_v := b_{T_v^k}$.

Proof: The proof is based on the induction of the function $J_v^{s_v}$ over nodes with increasing depths. The depth of a node v in $T_{v_1}^{s_{v_1}}$ is the length of the longest directed path (in terms of number of traversed edges) among all paths from v to a descendant of v without successor (e.g., the depth of node 2 in Figure 3 is two, and the depth of node 3 is zero). Let u be any node in $T_{v_1}^{s_{v_1}}$ with zero depth, then $J_u^0(x_u, b_{T_u^0})$, as defined in (3), is the optimal objective value of $P(T_u^0, x_u, b_{T_u^0})$ which is either 0 or ∞. This shows the base case of induction. Next, consider any node v with depth d, and assume:

H1: for every node u with depth less than d, for any $(x_u, b_{T_u^{s_u}}) \in B \times C$, the function value $J_u^{s_u}(x_u, b_{T_u^{s_u}})$ is the optimal objective value of $P(T_u^{s_u}, x_u, b_{T_u^{s_u}})$.

The induction is completed and the desired statement can be shown if for any $0 \le k \le s_v$, and for any $(x_v, b_{T_v^k}) \in B \times C$, the function value $J_v^k(x_v, b_{T_v^k})$ is the optimal objective value of $P(T_v^k, x_v, b_{T_v^k})$. The proof of this follows from another induction on the function J_v^k over k. For the base case, $J_v^0(x_v, b_{T_v^0})$ is the optimal objective value of $P(T_v^0, x_v, b_{T_v^0})$. Now assume another induction hypothesis:

H2: for any $0 \le l < k$, and for any $(x_v, b_{T_v^l}) \in B \times C$, the function value $J_v^l(x_v, b_{T_v^l})$ is the optimal objective value of $P(T_v^l, x_v, b_{T_v^l})$.

Let $w := S_v(k)$, subproblem $P(T_v^k, x_v, b_{T_v^k})$ is expressed into

$$\begin{aligned} \underset{\substack{x_u\,:\,u \in N(T_w^{s_w}) \cup N(T_v^{k-1}) \\ \sum_{u\,:\,\in N(T_w^{s_w})} g_u(x_u) + \sum_{u\,:\,\in N(T_v^{k-1})} g_u(x_u) \le b_{T_v^k}}}{\text{minimize}} & \left\{ F_{v,w} + \sum_{\substack{(i,j) \in E(T_w^{s_w}), \\ \text{or } (j,i) \in E(T_w^{s_w})}} F_{i,j}(x_i, x_j) \right. \\ & \left. + \sum_{\substack{(i,j) \in E(T_v^{k-1}), \\ \text{or } (j,i) \in E(T_v^{k-1})}} F_{i,j}(x_i, x_j) \right\}. \end{aligned} \tag{10}$$

Pulling out the minimization w.r.t. x_w and introducing an extra decision variable b_w, problem (10) can be written as

$$\begin{aligned} \underset{x_w, b_w}{\text{minimize}} \Bigg\{ & F_{v,w} + \underset{\substack{x_u\,:\,u \in N(T_w^{s_w}) \setminus \{w\} \\ \sum_{u\,:\,\in N(T_w^{s_w})} g_u(x_u) \le b_w}}{\text{minimize}} \left\{ \sum_{\substack{(i,j) \in E(T_w^{s_w}), \\ \text{or } (j,i) \in E(T_w^{s_w})}} F_{i,j}(x_i, x_j) \right\} \\ & + \underset{\substack{u\,:\,\in N(T_v^{k-1}) \\ \sum_{u\,:\,\in N(T_v^{k-1})} g_u(x_u) \le b_{T_v^k} - b_w}}{\text{minimize}} \left\{ \sum_{\substack{(i,j) \in E(T_v^{k-1}), \\ \text{or } (j,i) \in E(T_v^{k-1})}} F_{i,j}(x_i, x_j) \right\} \Bigg\}. \end{aligned} \tag{11}$$

Note that the second and third minimization problems in (11) are subproblems $P(T_w^{s_w}, x_w, b_w)$ and $P(T_v^{k-1}, x_v, b_{T_v^k} - b_w)$ respectively. Therefore, by the induction hypotheses H1 and H2 respectively, (11) can be rewritten as

$$\underset{x_w, b_w}{\text{minimize}} \left\{ F_{v,w} + J_w^{s_w}(x_w, b_w) + J_v^{k-1}(x_v, b_{T_v^k} - b_w) \right\}. \tag{12}$$

Thus, subproblem $P(T_v^k, x_v, b_{T_v^k})$ can be written as (12). In addition, according to (4) and (6) in **step 2**, $J_v^k(x_v, b_{T_v^k})$ is the optimal objective value of (12) with $(x_w^\star, b_w^\star)$ being an optimizing pair ($k-1$ and $b_{T_v^k}$ in (12) are, respectively, s_v and b_v in (4) and (6)). Therefore, the proof is completed. ■

As a consequence of Theorem 1, the optimal objective value of (1) can be obtained by a one-dimensional minimization of the cost-to-arrive function $J_{v_1}^{s_{v_1}}(\cdot, b)$.

Corollary 1: The value $\min_{x \in C} J_{v_1}^{s_{v_1}}(x, b)$ is the optimal objective value of the resource allocation problem in (1).

Proof: Theorem 1 states that $J_{v_1}^{s_{v_1}}(x, b)$ is the optimal objective value of $P(T_{v_1}^{s_{v_1}}, x, b)$ for any given $x \in C$ and b. Since the minimum optimal objective value of $P(T_{v_1}^{s_{v_1}}, x, b)$ with respect to $x \in C$ is the optimal objective value of (1), the desirable statement is obtained. ■

In addition to the optimal objective value, it is desirable to obtain an optimal solution to (1). The following statement certifies that **step 3** provides the desirable optimal solution:

Theorem 2: The procedure in **step 3** defines x_i^{opt} for each $i = 1, 2, \ldots, n$ exactly once. In addition, assuming that (1) is feasible, x_i^{opt}, $i = 1, 2, \ldots, n$ form an optimal solution to (1).

Proof: The fact that $T_{v_1}^{s_{v_1}}$ is a tree ensures that the BFS in **step 3** explores each node exactly once. Hence, x_i^{opt} for each $i = 1, 2, \ldots, n$ is defined exactly once. Let $v_2, v_3, \ldots, v_q$ denote all successors of v_1 with the order visited by the BFS procedure. By Corollary 1 and the definition of $x_{v_1}^{\text{opt}}$, the optimal objective value of (1) equals

$$\begin{aligned} & \min_{x \in C} J_{v_1}^{s_{v_1}}(x, b) \\ = \; & J_{v_1}^{s_{v_1}}(x_{v_1}^{\text{opt}}, b) \\ = \; & F_{v_1,v_2}(x_{v_1}^{\text{opt}}, x_{v_2}^{\text{opt}}) + J_{v_2}^{s_{v_2}}(x_{v_2}^{\text{opt}}, b_{v_2}^{\text{opt}}) + J_{v_1}^{(s_{v_1}-1)}(x_{v_1}^{\text{opt}}, b - b_{v_2}^{\text{opt}}). \end{aligned}$$

The second equality above is due to (7) and (9). Similarly, the term $J_{v_1}^{(s_{v_1}-1)}(x_{v_1}^{\text{opt}}, b - b_{v_2}^{\text{opt}})$ can be expanded into

$$\begin{aligned} & J_{v_1}^{(s_{v_1}-1)}(x_{v_1}^{\text{opt}}, b - b_{v_2}^{\text{opt}}) \\ = \; & F_{v_1,v_3}(x_{v_1}^{\text{opt}}, x_{v_3}^{\text{opt}}) + J_{v_3}^{s_{v_3}}(x_{v_3}^{\text{opt}}, b_{v_3}^{\text{opt}}) + J_{v_1}^{(s_{v_1}-2)}(x_{v_1}^{\text{opt}}, b - b_{v_2}^{\text{opt}} \\ & - b_{v_3}^{\text{opt}}). \end{aligned}$$

Expand $J_{v_1}^{(s_{v_1}-2)}(x_{v_1}^{\rm opt}, b-b_{v_2}^{\rm opt}-b_{v_3}^{\rm opt})$ and continue until

$$\begin{aligned} & J_{v_1}^1(x_{v_1}^{\rm opt}, b-b_{v_2}^{\rm opt}-\ldots-b_{v_{q-1}}^{\rm opt}) \\ = \; & F_{v_1,v_q}(x_{v_1}^{\rm opt}, x_{v_q}^{\rm opt}) + J_{v_q}^{s_{v_q}}(x_{v_q}^{\rm opt}, b_{v_q}^{\rm opt}) + J_{v_1}^0(x_{v_1}^{\rm opt}, b-b_{v_2}^{\rm opt}- \\ & \ldots - b_{v_q}^{\rm opt}). \end{aligned}$$

This leads to a decomposition of $J_{v_1}^{s_{v_1}}(x_{v_1}^{\rm opt}, b)$ into

$$\begin{aligned} & J_{v_1}^{s_{v_1}}(x_{v_1}^{\rm opt}, b) \\ = \; & \Big(F_{v_1,v_2}(x_{v_1}^{\rm opt}, x_{v_2}^{\rm opt}) + F_{v_1,v_3}(x_{v_1}^{\rm opt}, x_{v_3}^{\rm opt}) + \ldots \\ & + F_{v_1,v_q}(x_{v_1}^{\rm opt}, x_{v_q}^{\rm opt})\Big) + \Big(J_{v_2}^{s_{v_2}}(x_{v_2}^{\rm opt}, b_{v_2}^{\rm opt}) + \ldots \\ & + J_{v_q}^{s_{v_q}}(x_{v_q}^{\rm opt}, b_{v_q}^{\rm opt})\Big) + J_{v_1}^0(x_{v_1}^{\rm opt}, b-b_{v_2}^{\rm opt}-\ldots-b_{v_q}^{\rm opt}). \end{aligned} \tag{13}$$

Since (1) is assumed to be feasible, $\min_{x\in C} J_{v_1}^{s_{v_1}}(x,b) < \infty$. Then (3) implies that $J_{v_1}^0(x_{v_1}^{\rm opt}, b-b_{v_2}^{\rm opt}-\ldots-b_{v_q}^{\rm opt}) = 0$ and $g_{v_1}(x_{v_1}^{\rm opt}) + b_{v_2}^{\rm opt} + \ldots + b_{v_q}^{\rm opt} \le b$. Therefore, (13) simplifies to

$$\begin{aligned} & J_{v_1}^{s_{v_1}}(x_{v_1}^{\rm opt}, b) \\ = \; & \Big(F_{v_1,v_2}(x_{v_1}^{\rm opt}, x_{v_2}^{\rm opt}) + F_{v_1,v_3}(x_{v_1}^{\rm opt}, x_{v_3}^{\rm opt}) + \ldots \\ & + F_{v_1,v_q}(x_{v_1}^{\rm opt}, x_{v_q}^{\rm opt})\Big) + \Big(J_{v_2}^{s_{v_2}}(x_{v_2}^{\rm opt}, b_{v_2}^{\rm opt}) + \ldots \\ & + J_{v_q}^{s_{v_q}}(x_{v_q}^{\rm opt}, b_{v_q}^{\rm opt})\Big), \end{aligned} \tag{14}$$

with

$$g_{v_1}(x_{v_1}^{\rm opt}) + b_{v_2}^{\rm opt} + \ldots + b_{v_q}^{\rm opt} \le b. \tag{15}$$

Again by (7) and (9), for each $k = 2,3,\ldots,q$ s.t. $s_{v_k} > 0$ the term $J_{v_k}^{s_{v_k}}(x_{v_k}^{\rm opt}, b_{v_k}^{\rm opt})$ can be further expanded in a fashion similar to (14) and (15). The expansion is applied recursively until the scenario where if $J_u^{s_u}$ appears in the expression then $s_u = 0$. In the end, $J_{v_1}^{s_{v_1}}(x_{v_1}^{\rm opt}, b)$ is decomposed into

$$\begin{aligned} J_{v_1}^{s_{v_1}}(x_{v_1}^{\rm opt}, b) \; = \; & \sum_{i,j \,\big|\, (i,j)\in E(T_{v_1}^{s_{v_1}}) \text{ or } (j,i)\in E(T_{v_1}^{s_{v_1}})} F_{i,j}(x_i^{\rm opt}, x_j^{\rm opt}) \\ & + \sum_{i\in N(T_{v_1}^{s_{v_1}}): s_i = 0} J_i^0(x_i^{\rm opt}, b_i^{\rm opt}), \end{aligned} \tag{16}$$

with the additional fact that

$$g(x_v^{\rm opt}) + \sum_{k=1}^{s_v} b_{v_{S_v(k)}}^{\rm opt} \le b_v^{\rm opt} \quad \forall\, v \in N(T_{v_1}^{s_{v_1}}) : s_v > 0. \tag{17}$$

Summing up all inequalities in (17) leads to

$$\sum_{i\in N(T_{v_1}^{s_{v_1}})} g_i(x_i^{\rm opt}) \le b_{v_0}^{\rm opt} = b. \tag{18}$$

(16) and (18) together imply that $x_i^{\rm opt}$ for $i = 1,2,\ldots,n$ define an optimal solution to the resource allocation problem in (1). ■

The computation effort of the DP approach is dominated by the minimization of (4) in the **step 1/step 2** loop. For each node w being absorbed, (4) needs to be solved for each pair $(x_v, b_v) \in C \times B$, where v is the predecessor of w. This accounts for $O(|B|^2|C|^2)$ comparisons. Since each node in H is absorbed once except for v_1. The overall computation effort is $O((n-1)|B|^2|C|^2) = O(m|B|^2|C|^2)$. Therefore, it is established that problem (1) with tree interaction graph can still be solved with computation effort polynomial in the number of decision variables.

Corollary 2: With tree (or forest) interaction graph, the resource allocation problem in (1) can be solved with $O((n-1)|B|^2|C|^2) = O(m|B|^2|C|^2)$ comparisons.

Proof: Theorem 1 and Theorem 2 assert the correctness of the presented algorithm to solve (1). Since the algorithm requires $O((n-1)|B|^2|C|^2) = O(m|B|^2|C|^2)$ comparisons, the statement is obtained. ■

Remark 1: While Corollary 2 states that problem (1) with tree or forest interaction graph can be solved in polynomial-time in the number of decision variables and constraints, it does not mean that problem (1) can be solved in polynomial-time in problem size, since $|B|$ and $|C|$ need not be bounded from above by any polynomial function in problem size.

In certain cases, the computation effort for the minimization in (4) can be reduced. For instance, if $s_w = 0$ (i.e., w is a leaf node in the interaction graph H), then

$$(4) \Leftrightarrow \underset{x_w\in C}{\text{minimize}} \quad F_{vw}(x_v, x_w) + J_v^{s_v}\big(x_v, b_v - g_w(x_w)\big).$$

Similarly, if $s_v = 0$ (i.e., v has never absorbed any other node before), then

$$(4) \Leftrightarrow \underset{x_w\in C}{\text{minimize}} \quad F_{vw}(x_v, x_w) + J_w^{s_w}\big(x_w, b_v - g_v(x_v)\big).$$

In both cases, as well as the joint event, the computation cost for minimizing (4) reduces to $O(|B||C|^2)$ comparisons. In the description of **step 0** and **step 2-3** there is no mentioning of which leaf node to be chosen in case there are multiple leaf nodes. The choice of the leaf nodes affects the corresponding values of s_v and s_w in (4), when the leaf nodes are absorbed. This in turn affects the computation effort of DP. See Figure 4 for an illustration. It is possible to optimize over the order of absorbing leaf nodes in order to minimize the computation effort for DP. This, however, is not pursued in this paper.

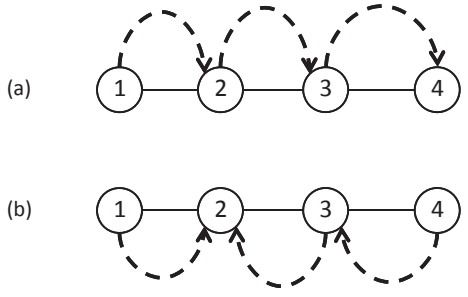

Fig. 4. Subfigure (a): the order of absorbed nodes is 1, 2, 3. Each time a node is absorbed, in the corresponding minimization in (4) $s_v = 0$. With this order of absorption, DP costs $O(|B||C|^2)$ comparisons. Subfigure (b): the order of absorption is 1, 4, 3. When node 3 is absorbed into node 2, both $s_v = 1$ and $s_w = 1$. Therefore, DP costs $O(|B|^2|C|^2)$ units of comparisons in this case. This figure demonstrates that the order in which the leaf nodes are absorbed can affect the computation cost for DP.

III. Dynamic Programming with Arbitrary Interaction Graph

The DP approach described in Section II does not apply to the instances of (1) where the interaction graphs contain cycles. For instance, consider the minimization of

$$\begin{aligned} \underset{x_1,x_2,x_3}{\text{minimize}} \quad & F_{12}(x_1,x_2)+F_{23}(x_2,x_3)+F_{31}(x_1,x_3) \\ \text{subject to} \quad & g_1(x_1)+g_2(x_2)+g_3(x_3)\le b. \end{aligned} \tag{19}$$

The interaction graph of this problem is a 3-cycle. The initialization **step 0** in the DP algorithm in Section II fails to find any starting leaf node that can be absorbed in **step 2**. However, if problem (19) is posed as a sequential optimization problem as follows:

$$\begin{aligned} \underset{x_1}{\text{minimize}} \quad & \Big(\min_{x_2,x_3} \underbrace{F_{12}(x_1,x_2)+F_{23}(x_2,x_3)+F_{31}(x_1,x_3)}_{:=h_{23}(x_2,x_3\,|\,x_1)}\Big) \\ \text{subject to} \quad & g_1(x_1)+\Big(g_2(x_2)+g_3(x_3)\Big)\le b, \end{aligned} \tag{20}$$

where $h_{23}(x_2,x_3\,|\,x_1)$ denotes a function of x_2 and x_3 when x_1 is given, then the inner optimization problem (with respect to x_2 and x_3) can be solved using the DP approach in Section II, because the interaction graph associated with the inner problem is a tree with two nodes (considering h_{23}). The process of pulling out the optimization with respect to x_1 and forming the inner problem with x_2 and x_3 has an interaction graph interpretation. Pulling out x_1 amounts to deleting node 1 and all edges connected to node 1 from the interaction graph in (19). By removing enough nodes from the interaction graph, any arbitrary interaction graph can be made acyclic. In general, with any permutation $i_1,i_2,\ldots,i_n$ of $1,2,\ldots,n$, problem (1) for any interaction graph can always be solved sequentially by

$$\begin{aligned} \underset{x_{i_1}}{\text{minimize}} \quad & \Big(\underset{x_{i_2}}{\text{minimize}} \quad \cdots \quad \Big(\underset{x_{i_r},\ldots,x_{i_n}}{\text{minimize}} \sum_{k=1}^{m} F_k\Big)\Big), \\ \text{s.t.} \quad & g_{i_1}(x_{i_1})+\Big(g_{i_2}(x_{i_2})+\ldots+\Big(g_{i_r}(x_{i_r})+\ldots\Big)\Big)\le b, \end{aligned} \tag{21}$$

where the innermost problem with respect to $x_{i_r},\ldots,x_{i_n}$ is associated with an acyclic interaction graph, and hence can be solved using the DP approach in Section II. However, the minimization with respect to $x_{i_1},x_{i_2},\ldots,x_{i_{r-1}}$ must be carried out by computationally intensive enumerations. Therefore, the fundamental problem regarding the minimization of the computation effort of (21) is to minimize the number variables $x_{i_1},x_{i_2},\ldots,x_{i_{r-1}}$ which require enumerations. In terms of interaction graph, the problem is to find the minimum cardinality subset of nodes to remove in order to render the remaining graph acyclic. This problem is well studied, and is known as the minimum feedback vertex set problem (MFVS problem) [11]. Even though the MFVS problem is NP-hard [12], numerous exact and approximation algorithms are available (e.g., [11]).

Using the proposed MFVS based organization of computation, problem (21) can be solved in $O(|B|^2|C|^{|V_{\text{mfvs}}|+2})$ units of basic computation, where $|V_{\text{mfvs}}|$ denotes the cardinality of the minimum feedback vertex set of the interaction graph. The characterization of computation effort in terms of the well-studied quantity $|V_{\text{mfvs}}|$ is a main advantage over the original nonserial DP approach in [1]. The computation effort of the approach in [1] is described through the optimal solution of a secondary optimization problem, which is not trivial to solve. In addition, using the proposed MFVS based approach to organize the computation in (21) appears to result in the same computation effort (in terms of asymptotic analysis) as in the method described in [1]. Figure 5 shows the interaction graph of an example instance of (1), for which the proposed method and the one in [1] require the same computation effort to solve (1) without the budget constraint.

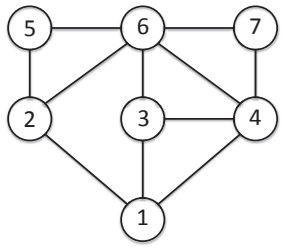

Fig. 5. An instance of interaction graph with seven nodes. The example is originally from [1]. A minimum feedback vertex set is $\{2,4\}$. This results in $O(|C|^{2+2})$ computation effort in solving (21) without budget constraint. This requires the same effort as that in [1].

IV. Conclusions

With the current result it is now evident that, in addition to the serial case where the interaction graph of the problem is a line, the more general resource allocation problem with tree or forest interaction graph can still be solved by DP with computation effort polynomial in the number of decision variables. Though this does not mean polynomial-time in problem size, since the numbers of quantization steps for the decision variable value and budget axises need not be bounded from above by any polynomial function in problem size. Also, it appears that the nonserial DP procedure introduced in [1] can be modified, resulting in a procedure whose computation effort is similar to that of the one in [1]. Yet the computation effort of the modified procedure can be characterized by the well-known minimum feedback vertex set problem instead of the secondary optimization problem required by [1]. Though a more detailed study is needed to examine the full potential of the proposed nonserial DP approach.

Appendix A

Given an instance of (2), a corresponding instance of (1) can be constructed as follows: the decision variables t_i's become x_i's. The resource budget constraint remains the same except for the change of the symbols of the decision variables. The interaction graph H for (1) is the undirected version of the time precedence graph G for (2). If in G two nodes are connected by two separate edges

in different directions, then in H there is only one edge connecting the two nodes. For each $\{i,j\}$ where a time precedence constraint $l_{ij} \leq t_j - t_i \leq u_{ij}$ is present in (2), an indicator function $I_{ij}(t_i,t_j)$ can be defined such that $I_{ij}(t_i,t_j) = 0$ if $l_{ij} \leq t_j - t_i \leq u_{ij}$, otherwise $I_{ij}(t_i,t_j) = \infty$. The integer m (i.e., the number of objective function components F_k's) is the number of time precedence constraints plus the number of "isolated" decision variables (i.e., decision variables not involved in any time precedence constraint). For each isolated decision variable x_i, there is a corresponding k with $F_k(x_i) = f_i(x_i)$. For each time precedence constraint $l_{ij} \leq t_j - t_i \leq u_{ij}$ involving nodes i and j, the corresponding F_k is defined as $F_k(x_i,x_j) = I_{ij}(x_i,x_j) + \frac{1}{\deg(i)} f_i(x_i) + \frac{1}{\deg(j)} f_j(x_j)$, where $\deg(i)$ and $\deg(j)$ are, respectively, the degrees of nodes i and j in the interaction graph H. By construction, a feasible choice of $t_1,t_2,\ldots,t_n$ in (2) corresponds to a choice of $x_1,x_2,\ldots,x_n$ in (1) with the same objective function value. On the other hand, an infeasible choice of $t_1,t_2,\ldots,t_n$ in (2) leads to a choice of $x_1,x_2,\ldots,x_n$ in (1) which is either infeasible or results in infinite objective function value. Therefore, it suffices to consider solving (1) instead of (2).

Finally, note that there exists instances of (1) that cannot be stated as instances of (2). For example, if F_k's in (1) involve products of decision variables, then this instance of (1) cannot be an instance of (2).

APPENDIX B

To see the claims, notice that each X_k defines a clique in H. Any clique with more than two nodes leads to cycles, contradicting the assumption that H is a tree. This implies that $|X_k| \leq 2$. On the other hand, $|X_k| = 1$ means that either there is another $X_{k'}$ such that $X_k \subset X_{k'}$, or H has an isolated node. The first case means that the X_k can be assimilated into the corresponding $X_{k'}$, and the second case is not allowed because H is assumed to be connected. Therefore, after all redundant X_k's have been processed, each set X_k has exactly two members. By definition, these two members correspond to two end nodes of an edge in the interaction graph H. This fact implies (b). Since each X_k requires an edge in H to match and there are no more than $n-1$ edges in a tree with n nodes, it holds that $m \leq n-1$. To complete the claim in (a), it suffices to notice that each edge in H also requires a set X_k (for some k) to match. Hence, $m \geq n-1$ and (a) is shown.

REFERENCES

[1] U. Bertele and F. Brioschi, *Nonserial Dynamic Programming*. Orlando, FL, USA: Academic Press, Inc., 1972.

[2] T. Ibaraki and N. Katoh, *Resource Allocation Problems: Algorithmic Approaches, no. 4 in Foundations of Computing Series*. The MIT Press, Cambridge, MA, 1988.

[3] K. C. Sou, H. Sandberg, and K. H. Johansson, "Nonserial dynamic programming with applications in smart home appliances scheduling – part I: Precedence graph simplification," preprint, available at the first author's homepage, 2013. [Online]. Available: http://www.math.chalmers.se/~cheong/NSDPpart1.pdf

[4] K. C. Sou, J. Weimer, H. Sandberg, and K.H. Johansson, "Scheduling smart home appliances using mixed integer linear programming," in *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*, dec. 2011, pp. 5144 –5149.

[5] K.C. Sou, M. Kördel, J. Wu, H. Sandberg, and K.H. Johansson, "Energy and co2 efficient scheduling of smart home appliances," in *European Control Conference*, 2013.

[6] A. Esser, A. Kamper, M. Frankje, D. Most, and O. Rentz, "Scheduling of electrical household appliances with price signals," in *Operation Research Proceedings*, 2006, pp. 253–258.

[7] T. Bapat, N. Sengupta, S. K. Ghai, V. Arya, Y. B. Shrinivasan, and D. Seetharam, "User-sensitive scheduling of home appliances," in *Proceedings of the 2nd ACM SIGCOMM workshop on Green networking*, 2011, pp. 43–48.

[8] N. Gatsis and G. Giannakis, "Residential demand response with interruptible tasks: Duality and algorithms," in *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*, dec. 2011.

[9] P. Du and N. Lu, "Appliance commitment for household load scheduling," *Smart Grid, IEEE Transactions on*, vol. 2, no. 2, pp. 411–419, 2011.

[10] G. Xiong, C. Chen, S. Kishore, and A. Yener, "Smart (in-home) power scheduling for demand response on the smart grid," in *Innovative Smart Grid Technologies (ISGT), 2011 IEEE PES*, 2011, pp. 1–7.

[11] F. Fomin, S. Gaspers, A. Pyatkin, and I. Razgon, "On the minimum feedback vertex set problem: Exact and enumeration algorithms," *Algorithmica*, vol. 52, no. 2, pp. 293–307, 2008.

[12] R. Karp, "Reducibility among combinatorial problems," in *50 Years of Integer Programming 1958-2008*, M. Junger, T. M. Liebling, D. Naddef, G. L. Nemhauser, W. R. Pulleyblank, G. Reinelt, G. Rinaldi, and L. A. Wolsey, Eds., 2010, pp. 219–241.