Optimal Capacity Expansion and Contraction
under Demand Uncertainty

Metin Çakanyıldırım *
School of Management, University of Texas at Dallas
Robin O. Roundy
School of Operations Research and Industrial Engineering, Cornell University
Samuel C. Wood
Graduate School of Business, Stanford University

November 23, 2001

Abstract

This paper presents a novel approach to compute optimal machine capacity expansion/contraction times under uncertain demand. A polynomial time Expansion/Contraction (EC) algorithm is developed to jointly solve the expansion and contraction problems when the demand is first stochastically increasing and then stochastically decreasing. The paper considers multiple machine types and allows for positive lead times for each type. It uses bottleneck policies (BP); Always buy machines of the bottleneck machine type to increase capacity and always retire machines in the reverse order. This order is used to optimally sequence machines types for expansion and contraction for regular service and capacity costs. The paper uses lost sales costs as a measure of the service. Capacity costs are computed through three components of machine specific costs: purchase and retirement costs that are independent of the usage, and machine rent that is proportional to the usage. EC algorithm is illustrated with a real life example drawn from the semiconductor industry.

*For correspondence: metin@utdallas.edu
1 Introduction

This paper models and illustrates an optimal solution strategy for a capacity expansion and contraction problem. A single product family experiences stochastic demand. The product family requires various operations on different machine groups. As demand increases new machines must be purchased to increase capacity. Conversely, machines are retired when the demand falls down.

Our model is motivated by equipment intensive industries, in which capacity expansion and contraction is critical and costly. For example, a new semiconductor fab costs around one billion dollars, and a single lithography machine costs around 4-5 million dollars. It is predicted that machine prices will continue to increase as products become more complex (known as Moore’s Law, Standard and Poor’s Industry Surveys [29] p.7). Finding the means to finance new capacity is a major problem. After the capacity is acquired, upper management naturally asks for high utilization. However machine purchase lead times are long (for example 6-12 months is common in the semiconductor industry [30] p.116), and future needs are affected by a variety of uncertainties. Consequently it is very important to buy and retire the correct set of machines at the correct times, so that demand matches productive capacity. The main purpose of this paper is to provide decision makers with a decision support tool to achieve such a match.

In many equipment intensive industries demand is volatile. Especially demand forecasts beyond the purchase lead times carry substantial uncertainty. The semiconductor industry and electric utilities are good examples. In the case of the semiconductor industry, for capacity planning purposes the most relevant demand figures are the ones extending from about 6 months to about 3 years into the future. The volatility of these figures are discussed in Çakanyıldırım and Roundy [8]. Due to rapid technology shifts there is a high risk that inventoried products will become obsolete. Consequently semiconductor companies carry only minimal inventories (Çakanyıldırım and Roundy [9]), and inventory provides only minimal protection against uncertainty. In the case of electric
utilities, short term capacity planning is mostly done daily against uncertain demand reaching its peak about early evenings. There can be substantial uncertainty in the hourly power demand. The product is a commodity and will not become obsolete. However inventories cannot be used to smooth out demand fluctuations because the storage is not efficient or economical. Summarizing, demand should be matched by regulating capacity because relying on inventory, except in the short term, is generally not economical.

We are assuming that all products have the same or very similar processing requirements, i.e. we have a single product family. In many industries a single product family is often composed of many different versions. When a new manufacturing technology becomes available, existing products are often migrated from an older technology to a newer one. New products undergo a verification process before they are ready for production. Thus the uncertainty in the total demand for a product family arises from several major sources - uncertainty in total marketplace demand, uncertainty in the company’s market share, uncertainty in the timing of migrations of a product from one technology to another, and uncertainty in the time when new products will be ready for production. We model the variability in the total demand for the product family, without differentiating between the different sources of uncertainty.

Customer demands generally cycle with the economic climate, increasing in economic booms and decreasing in economic slumps. Even high-tech industries which traditionally experience strong demand growth cannot escape from these cycles. A semiconductor professional once told: “… maybe … in the past 20-25 years we have had a 5-year period of growth, but I do not remember it. Two to three years-yes.” Therefore, capacity contractions must be studied along with expansions so we model both. Our model considers a finite time horizon, treats time as continuous variable, and models demand as a stochastic process. It does not permit any accumulation of inventory as motivated by the semiconductor industry and electric utilities, and unfilled demand is lost. Each machine type (or generator in electric utilities) has a lead time for purchase, installation and qualification. We assume that machine capacities remain constant over the decision horizon. We will next
discuss previous research and then the mathematical model will be presented. In section 4, solution procedure will be discussed and illustrated with a real life example. We provide a brief conclusion in sections 5.

2 Literature Survey

An extensive survey, capturing application areas and multi-location models, is in Luss [20], we discuss the models that appeared thereafter. Initially, capacity expansion work generally focused on models with deterministic demands. Neebe and Rao [22] provides a shortest path formulation and a Lagrangian relaxation scheme to sequencing and selecting expansion options. There have also been efforts to convert the stochastic expansion problem to an equivalent deterministic problem in the sense that both problems have the same optimal solutions. When product demands are transformed Brownian motion or transformed birth-death processes, Bean, Higle and Smith [4] provide an equivalent formulation by replacing stochastic demand with its deterministic counterpart and reducing the interest rate. Fong and Srinivasan [14] study a transportation problem where the capacity of facilities is expanded as the deterministic demand grows. Li and Tirupati [19] explicitly consider flexible and dedicated capacity expansions for multiple products and provide heuristics. Rajagopalan [24] presents a capacity model by assuming that a combination of machines exist whose total capacity is exactly equal to the demand. The model captures capacity expansions, disposals and replacements.

If time is a continuous variable rather than a discrete one, the expansion problem becomes an optimal control problem whose objective is usually the integral of cost (production, expansion, inventory) over time, e.g., Khmelnitsky and Kogan [18] provide an algorithm to calculate the optimal expansion rate function for deterministic demand. Davis, Dempster, Sethi and Vermes [11] regulate the expansion rate with an investment rate function. As soon as the cumulative investment reaches the random price of a discrete capacity unit, that unit becomes available. This paper enriches classical approaches by introducing a nonconstant investment rate and
random capacity prices which are more common in infrastructure industries than in manufacturing. Bena-
vides, Duley and Johnson [5] study the optimal expansion times for semiconductor fabs without differentiating between machine groups.

In the economics community, the capacity expansion problem is recently addressed by works of Dixit [12], and Eberly and Van Mieghem [13]. The latter establishes structural properties for expansion and contraction policies for multiple factors contributing to capacity. It provides a closed form solution of the optimal policy in the case of IID stochastic processes and stationary costs. It also introduces the concept of ordering expansions of different factors of capacity (see Proposition 3), which inspires the Bottleneck Policies of the current paper. Harrison and Van Mieghem [15] revisit Eberly and Van Mieghem [13] and study manufacturing costs explicitly. Their model is for discrete-time, continuous-capacity-expansion and multi-product case whereas the current paper proposes a model for continuous-time, discrete-capacity-expansion and single-product case. Angelus, Porteus and Wood [1] consider semiconductor fab capacity expansion with fixed costs, and stochastically increasing and correlated demand and apply inventory theory. They show that the optimal expansion policy is (s,S) type where both parameters depend on the most recently observed demand. Rocklin, Kashper and Varvaloucas [26] prove the optimality of (s,S) capacity expansion/contraction policies under certain conditions on the cost function and for a specific case (when demand exceeds the capacity, capacity is installed in an amount at least as large as the capacity deficit).

Rajagopalan, Singh and Morton [25] study the replacement of old vintage machines with new ones, under both certain and uncertain technology arrival times, and with deterministic, nondecreasing demand. They show some structural properties of the optimal solution and exploit those with a dynamic program. In competitive markets, companies pay attention to each other’s capacity expansions to avoid too much slack capacity. Using game-theoretic techniques, Bashyam [2] models a duopolistic market with a few stochastic demand scenarios under two cases. In the first (second) case, each company makes decisions without (with) the knowledge of the
other company’s decisions. The learning effect sometimes is not negligible, bringing per-unit costs and manufacturing times down. Under the learning effect, Hiller and Shapiro [16] provide a mixed integer programming formulation of capacity expansion where concave manufacturing costs are approximated by piecewise linear functions.

Although capacity expansions can be made at any time, quite a few models discretize time. Ong and Adams [23] examine the effects of granularity of time on cost for three cases: certain demand without shortage, certain demand with shortage, and uncertain demand. It is noted that the uncertain demand case is more sensitive to time granularity. The study advises using nonuniform granularity i.e., decisions are made more frequently in the short run than in the long run. In addition to making time discrete, another modeling practice is curtailing the decision horizon. It is valuable to know the shortest decision horizon such that the first period’s decisions do not change in response to events beyond that (decision) horizon. Bean and Smith [3] establish some criteria for the existence of a finite decision horizon and provide an algorithm to calculate it.

3 Multiple Machine Capacity Expansion and Contraction Model

In this section we will provide a mathematical description of the capacity expansion model that considers machine expansions and contractions. We consider a single product family. The family demand at time $t$ is $D_t(\omega) (\omega \in \Omega)$, a stochastic process over $[0, T]$ where $T$ is the length of the decision horizon.

We consider $M$ machine groups indexed by $i$, and we assume that all machines within a given group have the same capacity. If a machine of type $i$ is purchased at time $t$ then the machine is available at time $t + L(i)$. $L(i)$ is nonnegative machine installation lead time for machines in group $i$, it can include purchase and process set up/qualification lead times. After $t + L(i)$, the machine capacity is constant at $c_i$ units per time. When a machine is retired at time $t$, it immediately becomes unavailable at time $t$. We use machine purchases
(retirements) for capacity expansion (contraction) as demand fluctuates. Let \( n_i(t) \) (\( n_i(t) \geq 0 \)) represent the number of type-\( i \) machines available at \( t \). \( n_i(t) \) is the number of machines (of type \( i \)) existed initially plus those purchased by \( t \), minus those retired by \( t \). The overall capacity at time \( t \), \( K_t \), can be expressed as

\[
K_t = \min\{c_i \cdot n_i(t) : i = 1..M\}
\]

Thus \( K_t \) is a step function. Figure 1 depicts the capacity functions \( c_i \cdot n_i(t) \) for two machine groups, and a realization of the demand \( D_t \). The vertical bars in Figure 1 stand for the amount produced at time \( t \), i.e. \( \min\{D_t, K_t\} \).

We will now present our stochastic demand model. Let \( S \) be such that \( 0 \leq S \leq T \). We assume that the demand is stochastically increasing over \([0, S]\) and is stochastically decreasing over \([S, T]\). For the ease of exposition, we are considering the case where demand stochastically increases and decreases only once over \([0, T]\). Note that capacity expansions (purchases) happen during demand growth from 0 to \( S \) while contractions happen afterwards from \( S \) to \( T \). We can set \( S = 0 \) to model the situation where demand contracts without any growth.

We model two kinds of costs, capacity costs and lost-sales costs. Capacity costs include the cost of financing the purchase and installation of machines, and maintenance costs for the machines. We call capacity costs regular if postponing the purchase or earlier retirement of a machine does not increase them. This is typically the case.

The lost-sales cost measures service - the company’s ability to meet market demand. The measure we use in most of this paper is the expected value of the lost sales incurred during the time horizon \([0, T]\). Other service measures could be used, such as the expected number of weeks during which demand is fully met. We call a
service measure regular if it depends on the machine purchase/retirement schedule only through the capacity \( K_t \), and postponing the purchase or earlier retirement of a machine cannot decrease it. The measures described above are regular. We limit attention to regular service measures.

The installation of the \( k \)-th machine of type \( i \) at time \( t \) will raise the capacity of machine group \( i \) to \( a(i, k) := c_i n_i(t) \), recall that \( n_i(t) \) also accounts for existing machines. If this \( k \)-th machine is purchased at time \( t(i, k) - L(i) \), capacity goes up at the availability time \( t(i, k) \). Thus \( L(i) \leq t(i, k) \leq T \). For machines that are available initially, \( t(i, k) = 0 \). Similarly, the retirement of the \( k \)-th machine of type \( i \) at time \( u(i, k) \) will lower the capacity of machine group \( i \) from \( a(i, k) \) down to \( a(i, k - 1) \). Naturally, \( 0 \leq t(i, k) \leq u(i, k) \leq T \). If \( t(i, k) = u(i, k) \) then the \( k \)-th machine of type \( i \) is never purchased.

Let \( K \) be an upper bound on the capacity that we would consider installing before time \( S \). The set of machines, including the existing machines, \( \{(i, k) \colon a(i, k) < K\} \) is sorted in increasing order of \( a(i, k) \) and indexed by \( n, 1 \leq n \leq N \), so that \( a(i_n, k_n) := a_n \leq a_{n+1} \) and let \( t_n := t(i_n, k_n) \). Ties are broken arbitrarily. Define \( L_n = L(i_n) \) and \( u_n := u(i_n, k_n) \). We set \( t_n := 0 \) for all existing machines, \( t_N = u_N \), \( u_0 := T \) and \( a_N := K \). A bottleneck policy (BP) is a policy in which machines are made available for production in increasing order of \( n \) and are retired on a decreasing order of \( n \), i.e., \( t_n \leq t_{n+1} \) and \( u_{n+1} \leq u_n \).

**Lemma 1** If the machine purchasing problem has a regular cost function, a bottleneck policy minimizes the expected cost.

Proof: Suppose that we are given an instance of the machine purchasing problem and a set of machine availability and retirement times \( \{t_n, u_n : 1 \leq n \leq N, 0 \leq t_n \leq u_n \leq T\} \). First, let \( n \) be the smallest integer such that \( t_n > t_{n+1} \). We set \( t_{n+1} = t_n \). The capacity \( K_t, 0 \leq t < T \) is not effected. The capacity costs are regular so they do not increase, and for every sample path of \( D(t) \) the service cost does not change. Iterating
this procedure we put availability times in BP order without increasing the cost of the schedule. Now let \( n \) be the smallest integer such that \( u_{n+1} > u_n \) and set \( u_{n+1} = u_n \). Iterating this procedure, we can obtain a BP policy that does not cost more than the initial solution, for every sample path. □

The bottleneck policy can be implemented such that the most recently purchased machine is not retired first when demand starts falling. BP merely says that the most recently purchased machine and the first machine to retire must be of the same type. This distinction is important to avoid retiring new machines which may be slightly more efficient than older ones, although our model assumes that all characteristics of the machines of the same type are the same.

For now, we assume that the set of initially existing machines respect the BP, i.e., if \( n \) is an existing machines so are \( 1, 2, \ldots, n - 1 \). Let \( I \) be the existing machine with the largest BP index. We restrict attention to BPs. This determines the sequence in which machines are brought on line and are retired, but we still need to select the availability times \( t_n \) and retirement times \( u_n \), subject to the constraint

\[
0 = t_0 = t_1 = \ldots = t_I \leq t_{I+1} \leq t_{I+2} \ldots \leq t_N = u_N \leq \ldots u_2 \leq u_1 \leq u_0 = T.
\]

The capacity of the system over \([t_n, t_{n+1}) \cup (u_n, u_n] \) is \( a_n \), see Figure 1.

For our service measure we use \( S(t_1, \ldots, t_{N-1}, u_{N-1}, \ldots, u_1) \), defined as the expected value of the total unmet demand in \([0, T)\). Let \( n_{D_t}(a) := E[(D_t - a)^+] \), the expected amount by which the demand at time \( t \) exceeds \( a \). Then

\[
S(t_1, \ldots, t_{N-1}, t_N = u_N, u_{N-1}, \ldots, u_1) =
\]

\[
= \sum_{n=1}^{N} \left\{ \int_{t_{n-1}}^{t_n} n_{D_t}(a_{n-1})dt + \int_{u_{n-1}}^{u_n} n_{D_t}(a_n)dt \right\}
\]

\[
= \sum_{n=1}^{N} \left\{ \int_{t_{n-1}}^{t_n} \sum_{k=n}^{N} \left( n_{D_t}(a_{k-1}) - n_{D_t}(a_k) \right) + n_{D_t}(a_N)dt 
\right.+ \int_{u_{n-1}}^{u_n} \sum_{k=n}^{N} \left( n_{D_t}(a_{k-1}) - n_{D_t}(a_k) \right) + n_{D_t}(a_N)dt \right\}
\]

8
\[
\begin{align*}
\text{function. Actually by choosing } & n 	ext{ and subcontracting; Machine } \text{negative values but typically } G \\
\text{where } & \text{SC is a sunk cost, independent of the timing of machine purchases, we will not include it in our objective} \\
\text{We express the capacity costs as} & \text{function. Actually by choosing } K \text{ sufficiently large, SC can be brought down to zero. Note that the service} \\
\text{where} & \text{Since } SC \text{ is a sunk cost, independent of the timing of machine purchases, we will not include it in our objective} \\
\text{and} & \text{measure is a separable and additive function of } \{t_n, u_n : 1 \leq n < N\}. \\
\text{We express the capacity costs as} & \text{function. Actually by choosing } K \text{ sufficiently large, SC can be brought down to zero. Note that the service} \\
& \text{Since } SC \text{ is a sunk cost, independent of the timing of machine purchases, we will not include it in our objective} \\
& \text{measure is a separable and additive function of } \{t_n, u_n : 1 \leq n < N\}. \\
& \text{We express the capacity costs as} \\
& \text{where } G_n \text{ is the time-independent fixed cost (perhaps a portion) of buying and installing machine } n. \ G_n \text{ is} \\
& \text{incurred if the machine is purchased, i.e., } 0 < t_n < u_n. \ H_n \text{ is the salvage value. It can take positive or} \\
& \text{negative values but typically } |H_n| \leq G_n. \ H_n \text{ is incurred if the machine is bought and retired before } T, \text{i.e.,} \\
& t_n < u_n < T. \ h_n \text{ is an arbitrary constant. It captures usage } (u_n - t_n) \text{ dependent costs: such as the amortized} \\
& \text{cost of the capital (perhaps a portion of it) required to purchase and install the } n\text{th machine, plus the periodic} \\
& \text{maintenance cost. We coin the term machine rent for } h_n. \text{ Another interpretation of this cost structure is via} \\
& \text{subcontracting; Machine } n \text{ is subcontracted at } t_n \text{ by paying the fixed transaction cost } G_n. \text{ Machine is used} \\
& \text{until } u_n \text{ by paying a rent of } h_n \text{ per unit time. Subcontracting is terminated at time } u_n \text{ by incurring the fixed} \\
& \text{transition cost } H_n. \\
\end{align*}
\]
This cost structure can also be used to study a simplified version of Unit Commitment Problem ([28]) of power generators. Then machine purchases will correspond to turning the generators on and machine retirements correspond to turning the generators off. In order to apply this model to Unit Commitment Problem, a good sequence for the activation and the retirement of generators must be determined in advance. If that is not possible, this model can be used to evaluate optimal costs for given sequences, perhaps as a part of a (sequence) search heuristic. For now, we assume that $G_n = H_n = 0$ for all $n$. In the next section, we will discuss how to treat nonzero purchase costs and salvage values.

Recall that $L_n, 0 \leq L_n < T$ is the lead time required for purchase, installation and qualification of the $n$th new machine. Let $B(t \geq L) := \infty$ for $t < L$ and $B(t \geq L) := 0$ otherwise. We model the total cost associated with the purchase and retirement of $n$th machine as

$$f_n(t_n) := \eta_n(t_n) - h_n t_n + B(t_n \geq L_n), \quad 1 \leq n \leq N, \quad 0 \leq t_n \leq T$$

$$g_n(u_n) := \zeta_n(u_n) + h_n u_n, \quad 1 \leq n \leq N, \quad 0 \leq u_n \leq T \quad (1)$$

We use (1) in our computational study, but our theorems and algorithms do not require $f_n(.)$ or $g_n(.)$ to have any particular algebraic form. The machine capacity problem ($\mathcal{P}$) then becomes

$$\min \{ \sum_{n=1}^{N-1} f_n(t_n) + g_n(u_n) : 0 = t_0 = t_1 = \ldots = t_I \leq t_{I+1} \leq \ldots \leq t_{N-1} \leq t_N = u_N \leq u_{N-1} \leq \ldots \leq u_1 \leq u_0 = T \}$$

where $I$ is the largest index of exiting machines at $t = 0$. We break ties by favoring larger values of $t_n$. The next section describes the computation of optimal machine purchasing and retirement times $\{t_{I+1}, \ldots, t_{N-1}, u_{N-1}, \ldots u_1\}$. Note that capacity expansions (contractions) happen during demand growth (fall), thus without loosing any generality we can set

$$t_N = S = u_N. \quad (2)$$
Then we define expansion ($P^E$) and contraction ($P^C$) problems as

$$(P^E) \quad \min \left\{ \sum_{n=1}^{N-1} f_n(t_n) : 0 = t_0 = t_1 = \ldots = t_I \leq t_{I+1} \leq t_{N-1} \leq t_N = S \right\},$$

$$(P^C) \quad \min \left\{ \sum_{n=1}^{N-1} g_n(u_n) : S = u_N \leq u_{N-1} \leq \ldots \leq u_1 \leq u_0 = T \right\}.$$

Remark: In general there can be $K$ demand cycles, within each cycle demand stochastically increases or decreases only once. Then $[0, T]$ can be partitioned into $K$ expansion and contraction periods following each other; Expansion during $[T_{k-1}, S_k]$ and contraction during $[S_k, T_k]$ for $k = 1 \ldots K$ where $T_0 = 0$ and $T_K = T$. Note that most stochastic demand processes can be approximated fairly well by processes that stochastically increase and decrease several times over $[0, T]$. In addition, if fixed purchase and salvage costs are zero, a straight forward extension of ($P$) is possible by introducing $K$ copies of each availability time $t_n^k$ ($T_{k-1} \leq t_n^k \leq S_k$) and retirement time $u_n^k$ ($S_k \leq u_n^k \leq T_k$). Moreover, we can set $S_1 = 0$ to model the situation where demand contracts before growing.

Our algorithm for computing optimal availability times relies on the following assumption.

Assumption 1: $f_n(t)$ and $g_n(u)$ are convex functions which maps $[0, T]$ into $\mathbb{R} \cup \{+\infty\}$, for all $n, 1 \leq n \leq N$.

Under (1), the following lemma gives a sufficient condition for Assumption 1.

Lemma 2 If $D_t$ is stochastically increasing over $[0, S]$ and is stochastically decreasing over $[S, T]$, $\eta_n(t)$ and $\zeta_n(t)$ are convex respectively over $[0, S]$ and $[S, T]$.

Proof: It suffices to show that derivative of $\eta_n(t)$ and $\zeta_n(t)$ is nondecreasing in $t$.

$$\frac{d\eta_n(t)}{dt} = n_{D_t}(a_{n-1}) - n_{D_t}(a_n) = \int_{a_{n-1}}^{a_n} 1 - F_{D_t}(y) dy$$ (3)

where $F_{D_t}(.)$ is the cumulative density function of $D_t$. Since $D_t$ is stochastically increasing in $t$ ($0 \leq t \leq S$), the integrand is nondecreasing.
Similarly
\[
\frac{d\zeta_n(t)}{dt} = -n_D(t)(a_{n-1}) + n_D(t)(a_n) = \int_{a_{n-1}}^{a_n} -(1 - F_{D_t}(y))dy.
\] (4)

Since \(D_t\) is stochastically decreasing in \(t\) \((S \leq t \leq T)\), the integrand is nondecreasing. \(\Box\)

4 Calculation of Optimal Availability and Retirement Times

The capacity increment that results from installing machine \(n\) is \(a_n - a_{n-1}\). Since machines are usually of different types, the ratio of \(a_n - a_{n-1}\) to the cost of machine \(n\) can be very small or very large. In isolation, machines with small (large) capacity increment to cost ratio would be made available late (early). However, because of BP order, availability times of machines collide and certain sets of machines share the same availability time. Then the total capacity increment that results from making a set of machines simultaneously available becomes commensurate with the total cost of these machines. Similar comments can be made for machine retirements. Consequently, it is reasonable to expect that some machines are made available simultaneously while some others are retired simultaneously. We will call these group of machines clusters.

Let \(C\) denote a cluster of machines; We use clusters to model sets of machines that have the same availability times \(t_n\) or the same retirement times \(u_n\) in a solution to \((P)\). In view of \((P)\) and (2), capacity expansion and contraction problems can be solved entirely independently. Therefore, we do not need to concern ourselves with clusters that contain both the availability and the retirement of the \(N\)th machine. In the case of expansion, a cluster is a set of consecutive machines \(C := \{p, p+1, \ldots, q\}\), where \(1 \leq p \leq q \leq N\). Similarly for contraction, \(C := \{p, p-1, \ldots, q\}\), where \(N \geq p \geq q \geq 1\). We define \(\min(C) := \min\{n : n \in C\}\) and \(\max(C) := \max\{n : n \in C\}\). The root of cluster \(C\) is \(\min(C)\).

For convenience let us count machine retirements starting from \(N\) so that we can use the same index \(n\) for
machine purchases and retirements; \( n \) indicates a purchase (retirement) if \( n \leq N \) \((n > N)\). Recall that we are interested in clusters \( C \) such that either \( \max(C) \leq N \) or \( \min(C) > N \). Let

\[
f_C(t) := \begin{cases} 
\sum_{n \in C} f_n(t) & \text{if } \max(C) \leq N \\
\sum_{n \in C} g_n(t) & \text{if } \min(C) > N
\end{cases}
\]

The availability or retirement time associated with a given cluster \( C \) is computed by solving the following problem, called \((P_C)\).

\[
(P_C) \quad \begin{cases} 
\min\{f_C(t_C) : 0 \leq t_C \leq S\} & \text{if } \max(C) \leq N \\
\min\{f_C(t_C) : S \leq t_C \leq T\} & \text{if } \min(C) > N
\end{cases}
\]

**Assumption 2:** For each cluster \( C \) the optimal cost of \((P_C)\) is finite and \( f_C(t) \) has a unique minimizer.

**Lemma 3** Assumption 2 holds if both of the following holds:

i) For each \( y \), \( F_{D,t}(y) \) decreases strictly in \( t \) for \( 0 \leq t \leq S \).

ii) For each \( y \), \( F_{D,t}(y) \) increases strictly in \( t \) for \( S \leq t \leq T \).

Proof: By (3) and (4), \( f_n(t) \) and \( g_n(t) \) are strictly convex for each \( n \). This property is inherited by \( f_C(t) \).

Let \( t_C \) be the unique minimizer of \((P_C)\). More general versions of Assumption 2, Lemma 4 and Theorem 1 below appear in [10]. Similar versions are also in Jackson and Roundy [17], and Muckstadt and Roundy [21]. Best and Chakravarti [6], and Best, Chakravarti and Ubbaya [7] also study \((P)\) under the name isotonic regression. Since \((P^E)\) and \((P^C)\) are analogous problems, we will discuss the solution of \((P^E)\) only. Thus in Lemma 4 and Theorem 1 we deal with \((P^E)\).

**Lemma 4** For two nonempty disjoint clusters \( C_1 \) and \( C_2 \), if \( t_{C_1} \leq t_{C_2} \), then \( t_{C_1} \leq t_{C_1 \cup C_2} \leq t_{C_2} \).

Proof: Let \( C := C_1 \cup C_2 \). If \( t_{C_1} = S \) then \( S \) solves both \((P_{C_1})\) and \((P_{C_2})\), so it also solves \((P_C)\). Let \( t_{C_1} < S \). If \( t < t_{C_1} \) then Assumption 2 implies that \( f_{C_1}(t) > f_{C_1}(t_{C_1}) \) and \( f_{C_2}(t) \geq f_{C_2}(t_{C_1}) \). Thus \( f_C(t) > f_C(t_{C_1}) \), i.e.,
When $t > t_{C_2}$, since $t_{C_2} < S$ a similar argument shows that $t_C \leq t_{C_2}$. □

**Dual Feasibility Property:** A cluster $C$ has the Dual Feasibility Property if, whenever $C' := C \cap \{1, \ldots, n\} \neq \emptyset$ and $C'' := C \setminus C' \neq \emptyset$, we have $t_{C''} > t_{C'}$. Dual Feasibility and Lemma 4 imply that

$$t_{C''} \leq t_C \leq t_{C'}.$$  \hspace{1cm} (5)

If $J$ is a set of clusters that constitute a partition of $\{1, \ldots, N\}$, then $C(n)$ is the cluster in $J$ containing the $n$th machine. Thus $\min(C(n)) \leq n \leq \max(C(n))$ for all $n$.

**Theorem 1** Let $J$ be a partition of $\{1, \ldots, N\}$ into clusters, that has the following properties.

(i) (Primal feasibility): If $m < n$ then $t_{C(m)} \leq t_{C(n)}$.

(ii) (Dual feasibility): The Dual Feasibility Property holds for all $C \in J$.

Then if we set $t_n = t_{C(n)}$ for all $n$, we obtain an optimal solution to $(P_E)$

Proof: See the Appendix A.

An important consequence of Theorem 1 is that if we know where the breaks between clusters are, we can optimize each cluster separately by solving a problem of type $(P_C)$. From the definition of $(P)$, $(P^E)$, $(P^C)$ and (2), we can solve $(P)$ by solving $(P^E)$ and $(P^C)$ separately. Optimal clusters for $(P^E)$ and $(P^C)$ can then be put all together to construct a solution to $(P)$. We summarize this observation with a corollary.

**Corollary 1** Optimal solution to $(P)$ can be deduced from optimal clusters of $(P^E)$ and $(P^C)$.

Remember that we are using the same index $n$ for machine purchases and retirements. Let $R(r, s)$ be the roots of a set of clusters that give rise to an optimal solution to $(P_{r,s})$, where $(P_{r,s})$ is $(P)$ restricted to purchases or retirements in $\{r, r+1, \ldots, s\}$. We choose $r$ and $s$ such that either $r, s \leq N$ or $r, s > N$ so that $(P_{r,s})$ is a
smaller problem of type \((P^E)\) or \((P^C)\). We adapt the algorithm in section 3 of Muckstadt and Roundy [21] to solve \((P_{r,s})\). Our Cluster Algorithm is shown in Table I. Let the set of clusters deduced by \(R(r, s)\) be

\[
\{ C : \min(C) \in R(r, s) \text{ and } \max(C) + 1 \in R(r, s) \cup \{s + 1\} \}.
\]

Clusters deduced by \(R(r, s)\) of the Cluster Algorithm are the optimal clusters for \((P_{r,s})\) by Theorem 2.

- Table I -

**Theorem 2** The Cluster Algorithm produces an optimal solution to \((P_{r,s})\).

Proof: It suffices to show that the conditions of Theorem 1 are met by the clusters deduced by \(R(r, s)\). A simple induction proof establishes that these clusters are a partition of machine set \(\{r, \ldots, s\}\), and that the Primal feasibility property, holds. It suffices to prove that all clusters \(C\) that appear in any step of the algorithm have the Dual Feasibility Property. Singleton clusters automatically have this property, so it suffices to prove the following claim.

Claim: Let \(C_1\) and \(C_2\) both have the Dual Feasibility Property, let \(t_{C_1} > t_{C_2}\), let \(\max(C_1) = \min(C_2) - 1\), and let \(C := C_1 \cup C_2\). Then \(C\) has the Dual Feasibility Property.

Proof of Claim: Let \(C' := C \cap \{1, \ldots, n\}\) where \(n \in C\), and let \(C'' := C \setminus C' \neq \emptyset\). We need to show that \(t_{C'} > t_{C''}\). This holds by assumption if \(C' = C_1\). Assume that \(C' \subset C_1\). By Dual Feasibility \(t_{C'} > t_{C_1 \setminus C'}\), so by Lemma 4 and our assumptions, \(t_{C'} \geq t_{C_1} > t_{C_2}\). Note that \(C'' = (C_1 \setminus C') \cup C_2\). By Lemma 4, \(t_{C''}\) is between \(t_{C_1 \setminus C'}\) and \(t_{C_2}\), both of which are strictly less than \(t_{C'}\) so \(t_{C'} > t_{C''}\). The proof for \(C'' \subset C_2\) is similar. This proves both the claim and the theorem. \(\Box\)

**Theorem 3** The Cluster Algorithm takes \(O(N \cdot T^c)\) time to solve \((P_{1,N})\) or \((P_{N+1,2N})\), where \(T^c\) is the time required to solve a problem of the form \((P_C)\). If (1) holds then \(f_C(t)\) can be evaluated in time that is constant
in $|C|$, and $T^c = O(1)$.

Proof: Note that for either problem $(P_{1,N})$ or $(P_{N+1,2N})$, the number of roots $|R|$ starts with an initial value of $N$. It decreases by 1 with each Graft operation, is non-increasing as the Cluster Algorithm progresses, and is always positive. Consequently there can be at most $N - 1$ Graft operations, and each line of the Cluster Algorithm is executed at most $2N$ times. Each step of the algorithm takes at most $O(T^c)$ time, so the first assertion is proven.

Assume that (1) holds. Note that

$$\sum_{n \in C} \eta_n(t) = \int_{\tau=0}^{\tau=t} \{n_{D_r}(a_{\min(C)} - 1) - n_{D_r}(a_{\max(C)})\} d\tau \text{ if } \max(C) \leq N,$$

$$\sum_{n \in C} \zeta_n(t) = \int_{\tau=t}^{\tau=T} \{n_{D_r}(a_{\min(C)} - 1) - n_{D_r}(a_{\max(C)})\} d\tau \text{ if } \min(C) > N.$$

Thus $f_C(t)$ can be evaluated in time that is constant in $|C|$. Since $f_C(t)$ is convex and we are minimizing over a bounded interval, the minimization takes constant time. □

Examining the Cluster Algorithm, we see that we are solving a series of problems of type $(P_{1,s})$, $1 \leq s \leq N$, while solving $(P_{1,N})$. Then, it follows from Theorem 3 that the family problems of type $(P_{1,s})$, $1 \leq s \leq N$ can be solved in $O(N \cdot T^c)$. A similar argument yields that problems of type $(P_{N+1,s})$, $N < s \leq 2N$ can be solved in $O(N \cdot T^c)$. Consequently, all problems of type $(P_{r,s})$, $N < r \leq s \leq 2N$ can be solved in $O(N^2 \cdot T^c)$.

These observations show that steps A and B of the Expansion/Contraction (EC) algorithm in Table II can be completed in $O(N^2 \cdot T^c)$.

- Table II -

We discuss the validity of EC algorithm. Let $s$ be the index of the last machine installed and $r$ be the index of the last machine retired (counting retirement indices starting from $N + 1$). In order to retire a machine, it
must have been installed earlier, i.e., \(2N + 1 - s \leq r\). Suppose that there are no fixed purchase or salvage costs \((G_n = H_n = 0\) for all \(n\)). Let \(e_s\) be the cost of the expansion problem if \(s\) is the last machine installed. Let \(c_{2N+1-s,r}\) be the cost of the contraction problem where \(2N + 1 - s\) is the index of the first machine retired (by BP order \(s\) is the last machine installed) and \(r\) is the index of the last machine retired.

As promised earlier, now suppose that \(G_n\) and \(H_n\) are nonzero. In an optimal solution to \((P)\), naturally there is a last machine installed and a last machine retired. If we know the indices of these machines, \(s\) and \(r\), the optimal cost of \((P)\) is

\[
e_s + c_{2N+1-s,r} + \sum_{n=I+1}^{s} G_n + \sum_{n=2N+1-s}^{r} H_n \text{ where } I \leq s < N < 2N + 1 - s \leq r \leq 2N.
\]

(6)

If \(e_s\) and \(c_{2N+1-s,r}\) are available, this cost calculation can be done in \(O(N^2)\). Note that this is step C of EC Algorithm. Clearly step D takes only \(O(N \cdot T_c)\). In summary, EC Algorithm takes \(O(N^2 \cdot T_c)\). It also gives the optimal solution to \((P)\) because it searches over all possible \(s\) and \(r\) indices. Thus we arrive at our main theorem.

**Theorem 4** EC Algorithm solves \((P)\) in \(O(N^2T_c)\).

In the special case of all zero salvage values, the cost of \((P)\) becomes

\[
e_s + c_{2N+1-s,r} + \sum_{n=I+1}^{s} G_n \text{ where } I \leq s < N < 2N + 1 - s \leq r \leq 2N.
\]

While searching for \(r^*\), we consider only \(c_{2N+1-s,r}\), i.e. \(c_{2N+1-s,r^*} = \min\{c_{2N+1-s,r} : 2N + 1 - s \leq r \leq 2N\}\).

For the case of zero salvage values, let \(c_{2N+1-s,2N}\) be the optimal cost of the Contraction problem where \(2N + 1 - s\) is the index of the first machine to be retired or \(n\) is the last machine made available. In the case of zero salvage values, since the cost associated with \(c_{2N+1-s,r^*}\) is considered while finding \(c_{2N+1-s,2N}\), \(c_{2N+1-s,r^*} \geq c_{2N+1-s,2N}\). We can easily argue for \(c_{2N+1-s,r^*} \leq c_{2N+1-s,2N}\) to obtain \(c_{2N+1-s,r^*} = c_{2N+1-s,2N}\)
when salvage costs are zero. Then the cost of \((P)\) reduces to

\[ e_s + c_{2N+1-s} + \sum_{n=I+1}^{s} G_n \text{ where } I \leq s < N < 2N + 1 - s \leq 2N. \]

Thus, we can simplify Step B of EC Algorithm to solve the family of problems \((P_{2N+1-s,2N})\) for \(N < 2N + 1 - s \leq 2N\). A straightforward modification of the Cluster Algorithm solves this family of problems in \(O(N \cdot T^c)\). Thus Step B of EC Algorithm will take \(O(N \cdot T^c)\). We summarize these observation with a corollary below.

**Corollary 2** When salvage values are all zero, EC Algorithm can be streamlined to run in \(O(N \cdot T^c)\).

Up to now, we assumed that existing machines respect the BP order. We now discuss how to analyze the problem if that is not the case. Suppose that machines 1, \ldots, I exist initially as well as machine \(j\) where \(j > I + 1\) and there initially are no other machines. There are two possible actions: Either machine \(j\) is retired at time 0 or kept at least until \(S\); It can not be optimal to retire machine \(j\) at \(t\) where \(0 < t < S\). Suppose to the contrary that \(0 < t < S\), retiring \(j\) later we pay rent \(h_j\) and save lost sales costs. Since \(0 < t\), savings in lost sales must have balanced or exceeded the rent. However, because of stochastically increasing demand, lost sales savings will continue to exceed the rent as we delay \(t\) until \(S\). Consequently, \(t \geq S\).

In order to decide whether machine \(j\) should be retired immediately or kept until \(S\), it suffices to compare the costs of these options. If \(j\) is retired immediately, a cost of \(H_j\) is to be paid then the existing machines respect the BP order. The cost of this option is computed by adding \(H_j\) to costs in (6). When \(j\) is kept, we need to set \(a_{j-1} = a_j\) because as soon as \(j - 1\)st machine is installed the capacity becomes \(a_j\). We remove the \(j\)th machine from Expansion problem. Note that machine \(j\)'s rent is \(h_j(u_j - t_j)\), removing machine \(j\) from the Expansion problem causes \(t_j\) to drop from the cost expression. The remaining term \(h_ju_j\) reflects the true cost because \(u_j \geq S\) and machine \(j\)'s rent is paid during \([0, S]\) with this option. These operations effectively make the existing machines respect the BP order and we once more compute the costs by (6).
When there are more than one existing machine (say j and k) that destroy the BP order, the analysis above becomes more involved. It is not possible to evaluate retire and keep options for these machines separately machine by machine. In other words evaluation of whether to keep or retire machine j depends on whether machine k is kept or retired because different machines are coupled by \( a_{j-1} = a_j \) and \( a_{k-1} = a_k \) modifications suggested above. This forces us to consider all possible options together: retire j, retire k; retire j, keep k; keep j, retire k; keep j, keep k. Thus, evaluation of keep and retire options is not polynomial in the number of machines destroying the BP order. We close this discussion noting that the number of machines destroying the BP order is limited by the number of machine types and in most practical applications there will probably be at most 8-10 machines destroying the BP order.

We briefly comment on the size of the problem inputs i.e., how large \( N \) and \( T \) should be. With each machine we consider, the achievable capacity of the system increases. Since there is no point of installing capacity beyond the largest value \( D_S \) can take, an upper bound on \( N \) can be set as

\[
\bar{N} = \min \{ n : a_{n-1} > \sup_{\omega \in \Omega} D_S(\omega) \}.
\]

On the other hand, \( T \) denotes the end of the contraction period. \( T \) must be identified from demand forecasts so planners actually do not choose it. Identification of \( T \) is simple when demand forecasts indicate a single demand growth and fall cycle. When there are multiple growth and fall cycles, planners must choose how many cycles to consider. In the case of zero \( G_n \) and \( H_n \), cycles become independent. Then by the structure of \((P)\) instalment and retirement times in \([0, \tau] \) (\( 0 \leq \tau < T \)) are unaffected by considering additional cycles beyond \( T \); as far as the immediate decisions are concerned, it is sufficient to solve a single cycle problem. For nonzero \( G_n \) and \( H_n \) cycles affect each other and planners must make an effort to consider all the demand cycles.

Lastly, we illustrate the EC algorithm with real life data from the semiconductor industry. We obtained machine data from SEMATECH (SEmiconductor MAnufacturing TECHnology: www.sematech.org) databases.
Machine data includes purchase prices, capacities (in number of wafers per month) and delivery lead times (in months). There are about 40 machine types, each of which is necessary to manufacture a single wafer. Machine delivery lead times range from 12 months to 24 months. Purchase prices are between $0.4 M and $11 M. We assumed that fixed machine purchase costs and salvage values are zero, and set the lost sales costs at $1.2 K per wafer. Initial fab capacity is about 5900 wafers per month. We plan for capacity starting 12 months from now and ending 67 months from now, so lead times shorter than or equal to 12 months are irrelevant. At each month, demand is modelled as a trapezoid density with lower and upper bounds shown in Figure 2. Demand is stochastically increasing for the first 30 months ($S = 30$) and stochastically decreasing in the remaining 25 months ($T = 55$). We plan to buy at most 82 machines in the 55 month planning horizon ($N = 82$). According to the optimal solution shown in Figure 2, only 34 machines are installed in 5 different months (5 clusters) in the first 30 months of demand growth. In the remaining 25 months, 31 machines are retired in 5 different months (5 clusters) leaving a capacity of about 6600 wafers per month.

5 Conclusion

We provided a polynomial time algorithm to optimally plan for capacity expansions and contractions and illustrated its use with a real life example drawn from the semiconductor industry. We have assumed that a single product family experiences stochastic demand that first stochastically increases and then decreases. Machines are purchased while demand is growing and are retired while demand is falling. A good portion of the existing capacity expansion literature deals with single machine types. Multiple-machine type models are generally heuristics. Our EC Algorithm fills this gap providing an optimal solution to a multiple machine type problem by introducing BP order for the sequence of purchases and retirements. BP order remains optimal for
regular cost functions, in addition to our specific lost sales and capacity costs. We have discussed that capacity
cost structure is general enough to accommodate several interpretations: machine purchases (with or without
salvage values), subcontracting, simplified versions of unit commitment problem. As we remarked earlier, by
concatenating demand rise and fall periods one after another fairly general demand process can be modeled.
Achieving these with positive lead times add to the value of EC Algorithm.

Our model is motivated by equipment intensive industries such as the semiconductor industry and electric
utilities where capacity is costly and have considerable installation lead times. Capacity planning involves
investing large amounts of capital in the face of uncertainty. It is the primary irreversible decisions that have
long term effects on competitiveness and profitability. Hence it is desirable to make correct decisions far in
advance while accounting for large uncertainty in the demand. Capacity plans must consider the risks that
arise from uncertainty, which can only be built in via a general stochastic demand model. Considering the
capabilities of EC Algorithm and the fact that it is computationally manageable, it appears to be a useful tool
for capacity planning.

In order to apply the EC Algorithm to the unit commitment problem, the optimal order of generator instal-
lations and retirements must be known in advance. Future research on obtaining such an order is necessary. EC
algorithm deals with a single product family. Some work has been done by Roundy, Zhang and Çakanyıldırım
[27] to study multiple machine type expansions for multiple product families with uncertain demands under a
specific capacity shortage allocation (to product families) criterion.
Appendix A: Proof of Theorem 1

**Theorem 1** Let $J$ be a partition of $\{1, \ldots N\}$ into clusters, that has the following properties.

(i) (Primal feasibility): If $m < n$ then $t_C(m) \leq t_C(n)$.

(ii) (Dual feasibility): The Dual Feasibility Property holds for all $C \in J$.

Then if we set $t_n = t_C(n)$ for all $n$, we obtain an optimal solution to $(P^E)$.

Proof: Let $f^+_n(t)$ and $f^-_n(t)$ be the right-hand and left-hand derivatives of $f_n$ at $t$. Let $C_n^\leq := \{ k : k \leq n, k \in C(n) \}$. Let $f^\leq_n(t) := \sum_{k \in C_n^\leq} f^-_k(t)$ and let $f^-_n(t)$, $f^+_n(t)$ and $f^\leq_n(t)$ be similarly defined. Since $(P)$ is a convex program, the first-order conditions establish optimality. These conditions are

$$x_0 = 0.$$  

For $1 \leq n \leq N$, $f^+_n(t_n) \leq z_n \leq f^-_n(t_n)$, $z_n = x_{n-1} - x_n$, $t_{n-1} \leq t_n$, $x_{n-1} \geq 0$, and $(t_n - t_{n-1}) x_{n-1} = 0$. 

(7)

In two steps, we show that clusters in $J$ satisfying the conditions of the theorem solve (7). The first step is establishing that (7) holds whenever (8) has a solution for each cluster $C \in J$.

$$f^-_n(t_C) \leq z_n \leq f^+_n(t_C), \quad \forall n \in C, \quad \sum_{n \in C} z_n = 0, \quad \sum_{k \in C_n^\leq} z_k \leq 0, \quad \forall n \in C. \quad (8)$$

Suppose that $C \in J$ solve (8), we produce a solution to (7) as follows. Recall that $t_n = t_C$ for all $n \in C$. If $C(n - 1) \neq C(n)$ we set $x_{n-1} = 0$. Otherwise, we set $x_n = -\sum_{k \in C_n^\leq} z_k \geq 0$. This solves (7).

As the second step, we obtain a solution to (8) as follows. The definition of $t_C$ implies that $\sum_{k \in C} f^+_k(t_C) \geq 0 \geq \sum_{k \in C} f^-_k(t_C)$. Similarly, $\sum_{k \in C_n^\leq} f^+_k(t_{C_n^\leq}) \geq 0 \geq \sum_{k \in C_n^\leq} f^-_k(t_{C_n^\leq})$. By (5), $t_{C_n^\leq} \geq t_C$. By convexity, $n \in C$ implies

$$f^-_n(t_C) = \sum_{k \in C_n^\leq} f^-_k(t_C) \leq \sum_{k \in C_n^\leq} f^-_k(t_{C_n^\leq}) \leq 0. \quad \text{Similarly, } f^+_n(t_C) \geq 0. \quad (9)$$
Note that $f_{<n}(t_C) + f_{\geq n}^+(t_C)$ is a non-increasing function of $n$, that it is non-negative for $n = \min(C)$, and that it is non-positive for $n = \max(C) + 1$. Thus there is an $n \in C$ such that

$$f_{<n}(t_C) + f_{\geq n}^+(t_C) \geq 0 \geq f_{\leq n}^-(t_C) + f_{>n}^+(t_C).$$  

(10)

We set $z_k = f_k^-(t_C)$ for $k < n$, $z_k = f_k^+(t_C)$ for $k > n$ and $z_n = -f_{<n}^-(t_C) - f_{>n}^+(t_C)$. Thus, $\sum_{n \in C} z_n = 0$. For $k < n$, using (9) $\sum_{l \in \langle k} z_l = f_{\leq k}^-(t_C) \leq 0$. For $k \geq n$, using (9) $\sum_{l \in \langle k} z_l = -\sum_{l \in \rangle k} z_l = -f_{>k}^+ \leq 0$. Finally, note that (10) implies $z_n - f_n^-(t_C) = -f_{<n}^-(t_C) - f_{>n}^+(t_C) \geq 0$ and that $f_n^+(t_C) - z_n = f_{\leq n}^+(t_C) + f_{\geq n}^-(t_C) \geq 0$. Thus (8) holds. \(\square\)

**Acknowledgment**: This work was supported by a grant from Semiconductor Research Cooperation under the task “Modeling Random Processes”.


References


Figure 1: Production capacity versus demand for a semiconductor fab.
Figure 2: Optimal capacity as demand grows and falls.
• INITIALIZE: $R := \{r, r + 1, \ldots, s\}$, $C = \{r\}$, $n := r + 1$

While $n < s + 1$ do

• SOLVE ($\mathcal{P}_{r,n}$):

  • $C' := \{n\}$, $GraftComplete := false$

  While $\min(C') > r$ and $GraftComplete = false$ do

    • $k := \min(C') - 1$

    If $t_{C'} < t_C(k)$ then

      • GRAFT: $R := R \setminus \min(C')$, $C' := C' \cup C(k)$

    else

      • $GraftComplete := true$

    endwhile

  • $n := n + 1$

endwhile

$R(r, s) := R$

Table I: Cluster Algorithm to solve ($\mathcal{P}_{r,s}$)
A: Solve Expansion Problems.

Use Cluster Algorithm to Solve \((\mathcal{P}_{I+1,s})\) for \(I \leq s < N\).

Record the cost \(e_s\).

B: Solve Contraction Problems:

for \(r = N + 1\) to \(2N\) do

Use Cluster Algorithm to Solve \((\mathcal{P}_{r,s})\) for \(r \leq s \leq 2N\).

Record the cost \(c_{r,s}\).

endfor

C: \((r^*, s^*) = \arg\min\{e_s + c_{2N+1-s,r} + \sum_{n=I+1}^{s} G_n + \sum_{n=2N+1-s}^{r} H_n : I \leq s < N < 2N + 1 - s \leq r \leq 2N\}\)

D: Find optimal availability and retirement times using the Cluster Algorithm on \((\mathcal{P}_{1,s^*})\) and \((\mathcal{P}_{2N+1-s^*,r^*})\)

Table II: EC Algorithm to solve \((\mathcal{P})\)