Optimal Machine Capacity Expansions with Nested Limitations 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

December 2, 2001

Abstract

This paper studies machine capacity expansions for a production facility facing uncertain customer demand. The capacity of the facility has nested and expandable limitations. Depending on the application, these limitations may represent shop floor space, building shell space, water reservoir capacity or environmental permits. The paper uses machine capacity costs that have two components: purchase costs that are independent of the usage and machine rent that is proportional to the usage. The cost of expanding a limitation depends on the current size of the limitation and the amount of expansion. The customer service is represented by the lost sales cost. The paper presents a polynomial time algorithm (FIFEX) to minimize the total costs by computing machine capacity expansion times jointly with the expansion times of limitations. It considers multiple machine types and allows for positive lead times for each type. Demand is assumed to be nondecreasing in a “weak” sense.

*For correspondence: metin@utdallas.edu
1 Introduction

We study an optimal capacity expansion strategy for a facility experiencing stochastic demand for a single product family. The product family requires various operations on different machine groups. New machines must be installed to raise the capacity of the facility as demand increases. There are nested limitations on the capacity however, these limitations can be expanded as well. Limitations may correspond to shop floor space, building shell space, water reservoir capacity, environmental permits, sewer connections, piping, etc. Inspired by, but not limited to, the modular floor and shell spaces expansion trends of the semiconductor industry ([5]), we will specifically call limitations on the capacity as “floor space” and “shell”. Floor space and shell may refer to physical spaces as well as other physical or abstract capacity limitations mentioned above.

Our model captures the cost of bringing existing floor space into productive use, and the cost of expanding the total shell. Both of these actions have associated costs that depend on the current capacity and the amount of capacity expansion. Also we have machine costs with two parts: purchase costs that are independent of the usage and machine rent that is proportional to the usage. In addition we have lost sales costs to measure the lack of customer service. The main contribution of this paper is in providing a stochastic model that minimizes total costs by jointly optimizing machine purchase decisions for all machine groups, and floor space and shell expansion decisions.

In equipment intensive industries, profitability depends heavily on machine capacity planning. It is very important that machines are installed in a timely manner to match the capacity with the demand. Since floor space and shell limit the number, size or capacity of the machines that can be installed, machine capacity must be planned jointly with floor space and shell space. Construction of a facility and instalment of machines is a very costly affair, for example new semiconductor fabs cost about $1-2 billion. According to [2], the semiconductor industry reinvested 23% of total revenue in capital expenses in 1996, about 60-70% of that went into tool purchases. [17] reports that 75% of new semiconductor fab expenditure is for tool purchases and continues to argue that expenditures for new tools are siphoning off manufacturers’ profits. Such high costs create a financing problem for manufacturers and increasing machine prices seem to highlight this problem.

The financing problem is complicated by long machine purchase and construction lead times, and demand volatility.
In the semiconductor industry lead times of 6-18 months for machines and 12-18 months for construction are common. Because of these lead times, the relevant demand forecasts for planning are those of about 1-5 years into the future. These forecasts generally have substantial uncertainty. For example, see [8] for the magnitude and discussion of demand uncertainty in the semiconductor industry. In an effort to minimize the risk of obsolete inventory due to demand uncertainty, semiconductor companies tend to carry small inventories [9]. As a result of these observations, our model will have a medium to long planning horizon and random demand. It will not permit the accumulation of inventory, and unfilled demand will be lost. We model capacity expansions in an expanding market and do not consider capacity contractions. In the context of the semiconductor industry the time horizon captures the capacity expansion phase for newer manufacturing technologies and the product maturity phase; it does not capture capacity phase out because capacity phase out is usually not as financially significant as capacity build up. Each machine type has a lead time for purchase, installation and qualification for manufacturing. We assume that once a machine is ready for production, its capacity will remain constant over the time horizon.

Although we have used arguments and numbers from the semiconductor industry to motivate our model, the model is general enough to be used by other industries. In fact we do not even mention the phrase semiconductor industry in our model development and analysis. In the next section we will discuss previous research. In section 3 the machine capacity expansion model will be presented mathematically. This will be followed by the introduction of shop floor and shell space expansions in section 4 and the presentation of FIFEX in section 5. We provide a real life example and a brief conclusion in sections 6 and 7, respectively.

2 Literature Survey

A detailed survey of capacity expansion models can be found in Luss [22]. Models that appeared afterwards are discussed in detail in [10]. For completeness, we briefly glance at the existing literature. Capacity can be expanded against deterministic or stochastic demand. In the first category we have: Neebe and Rao [23] providing a model to select and order capacity expansions. Bean, Higle and Smith [4] converting stochastic problems to deterministic ones

A general approach to capacity planning under uncertainty is stochastic programming (Wets [28], Birge and Louveaux [7]). Generally demand uncertainty is represented in terms of demand scenarios. Eppen, Martin and Schrage [15], Escudero, Kamesam, King and Wets [16] and Chen, Li and Tirupati [11] use scenarios in studying capacity planning in manufacturing. Takriti, Birge and Long [27] attack electricity generator on/off planning with scenarios. All these have indicator (integer) variables to represent discrete capacity augmentations. One of the strengths of the current paper is in avoiding integer variables. Other examples of continuous time models are Khmelnitsky and Kogan [20] and Davis, Dempster, Sethi and Vermes [12], both of which study the optimal expansion rate. There also have been attempts to apply inventory theory to capacity expansion problems to obtain structural results: Angelus, Porteus and Wood [1], and Rocklin, Kashper and Varvaloucas [26].

In the economics community, the capacity expansion problem is recently addressed by works of Dixit [13], and Eberly and Van Mieghem [14]. The latter introduces the concept of ordering expansions of different factors of capacity (see Proposition 3), which inspires the Bottleneck Purchasing Policies of the current paper. 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. A game theoretic capacity expansion model with two companies is given in Bashyam [3]. 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. Under the learning effect, Hiller and Shapiro [18] provide a mixed integer programming formulation of capacity expansion.

Benavides, Duley and Johnson [5] study the optimal capacity expansion times for semiconductor fabs. They talk about modular space expansions: “Sequentially deployable large fabs are . . . attractive since they offer the economies of scale of larger fabs but require a smaller initial capital outlay”. If demand is expected to grow rapidly, companies may take advantage of the strong economies of scale in shell space expansion by building a large shell and adding floor space in increments. However, [5] is an aggregate capacity model -not differentiating between machine groups.
Çakanyıldırım, Roundy and Wood [10] study optimal machine capacity expansion and contraction with uncertain demand but without considering floor or shell space. The current paper expands the ideas of [10] without losing optimality to include the modular space expansion concept of [5].

3 Multiple Machine Capacity Expansion Model

In this section we will provide a mathematical description and analysis of our model. Roughly speaking, our discussion is a specialization of [10] for expansion, except for some subtle generalizations in the expression of capacity costs, Assumption 2 and Lemma 2. We will state several results from [10] without proof before discussing floor and shell expansions.

We have $M$ 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 will be available a lead time $L(i)$ later at time $t + L(i)$. From then on its capacity is $c_i$ wafers per time (e.g. per week). Let $I(i)$ be the capacity of machine group $i$ at time 0, and let $n_i(t)$ represent the number of additional type-$i$ machines that are made available in $(0, t]$. The overall capacity at time $t$, $K_t$, is

$$K_t = \min \{I(i) + c_i * n_i(t) : i = 1..M\}$$

Thus $K_t$ is a nondecreasing step function. Figure 1 depicts the capacity functions $I(i) + c_i * n_i(t)$ 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. $\min\{D_t, K_t\}$.

– Figure 1 –

We have 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. The lost-sales cost measures service—the company's ability to meet market demand. We call capacity (lost sales) costs regular if postponing the purchase of a machine does not increase (decrease) them.
The installation of the $k$-th machine of type $i$ at time $t$ will raise the capacity of machine group $i$ to $a(i,k) := I(i) + c_i n_i(t)$. The $k$-th machine is purchased at time $t(i,k) - L(i)$, and capacity goes up at the availability time $t(i,k)$. Thus $L(i) \leq t(i,k) \leq T$. If $t(i,k) = T$ then the purchase of the $k$-th machine of type $i$ is deferred beyond the end of the time horizon. Let $\mathcal{K}$ be an upper bound on the capacity that we would consider installing before time $T$. The set of machines $\{(i,k) : a(i,k) < \mathcal{K}\}$ is sorted in increasing order of $a(i,k)$ and indexed by $n, 1 \leq n < N$, so that $a(i_n,k_n) := a_n \leq a_{n+1}$ and $t(i_n,k_n) := t_n$. Ties are broken arbitrarily. For convenience we set $t_0 := 0$, $a_0 := \min \{I(i) : i = 1 \ldots M\}$, $t_N := T$ and $a_N := \mathcal{K}$. A bottleneck purchasing policy (BPP) is a policy in which machines are made available for production in increasing order of $n$, i.e., $t_n \leq t_{n+1}$. We specialize the following Lemma from [10].

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

We restrict attention to BPPs. This determines the sequence in which machines are installed, but we still have to solve for the availability times $t_n$, subject to the constraint

$$0 = t_0 \leq t_1 \leq t_2 \leq \ldots \leq t_N = T.$$

The capacity of the system between time $t_n$ and time $t_{n+1}$ will be $a_n$, see Figure 1.

For a service measure we use $S(t_1, \ldots t_{N-1})$, defined as the expected value of the total demand lost in [0, T). Let $D_t$ be the stochastic demand at time $t$, $t \in [0,T)$. Let $n_{D_t}(a) := E[(D_t - a)^+]$, the expected amount by which the demand at time $t$ exceeds $a$. Then using some algebra ([10])

$$S(t_1, \ldots t_N) = \sum_{n=1}^{N} \int_{t_{n-1}}^{t_n} n_{D_t}(a_{n-1}) dt = \sum_{k=1}^{N-1} \eta_k(t_k) + SC$$

where $\eta_k(t_k) := \int_{t_{k-1}}^{t_k} (n_{D_t}(a_{k-1}) - n_{D_t}(a_k)) dt$ and $SC := \int_{t_{n-1}}^{t_N} n_{D_t}(C) dt + \eta_N(T)$. Since $SC$ is a sunk cost, independent of the timing of machine purchases, we will not include it in our objective function. Note that the service measure is a separable function of $\{t_n : 1 \leq n < N\}$. 

5
We express the capacity costs as
\[ \sum_{n=1}^{N-1} G_n \cdot 1_{(t_n < T)} + g_n(t_n). \]

\( G_n \) denotes the time-independent fixed cost of buying and installing machine \( n \). \( G_n \) is incurred if the machine is bought before \( T \); the function \( 1_{(t_n < T)} \) indicates that. \( g_n(.) \) is an arbitrary convex function. It captures time \( (t_n) \) dependent costs: such as the amortized cost of the capital (perhaps a portion of it) required to purchase and install the \( n \)th machine, plus the periodic maintenance cost. For now, we assume that \( G_n = 0 \) for all \( n \). At the end of this section, we will discuss how to handle nonzero fixed costs with the Cluster Algorithm. If \( t_n = T \) then the purchase of the \( n \)-th machine is deferred beyond the end of the time horizon. \( L_n, 0 \leq L_n < T \) is the installation lead time of the \( n \)th new machine. Let \( B(t \geq L) := \infty \) for \( t < L \) and \( B(t \geq L) := 0 \) otherwise. The total cost associated with the \( n \)th machine is
\[ f_n(t_n) := \eta_n(t_n) + g_n(t_n) + B(t_n \geq L_n), \quad 1 \leq n \leq N, \quad 0 \leq t \leq T \]

We use (1) in our computations, but our theorems do not require \( f_n(y) \) to be in any particular algebraic form. The machine purchasing problem \((P)\) then becomes
\[
(P) \quad \min \{ \sum_{n=1}^{N-1} f_n(t_n) : t_0 \leq t_1 \leq ... \leq t_{N-1} \leq t_N = T \}
\]

We break ties by favoring larger values of \( t_n \). The problem of type \((P)\) was studied in [19] and in [6] under the name isotonic regression.

Having defined the problem, we will now propose a solution method. Our method for computing optimal availability times relies on the following assumptions.

**Assumption 1:** \( f_n(t) \) is a convex function which maps \([0, T]\) into \( \mathbb{R} \cup \{+\infty\} \), for all \( n, 1 \leq n \leq N \).

Note that under (1), if \( D_t \) is stochastically increasing in \( t \), then Assumption 1 holds.

A cluster is a set of consecutive machines \( C := \{p, p+1,..., q\} \), where \( 1 \leq p \leq q \leq N \). We use clusters to model sets of machines that have the same availability times \( t_n \) in a solution to \((P)\). We define \( \min(C) := \min\{n : n \in C\} \) and \( \max(C) := \max\{n : n \in C\} \). The root of cluster \( C \) is \( \min(C) \). Let \( f_C(t) := \sum_{n \in C} f_n(t) \). The availability time
associated with a given cluster $C$ is computed by solving the following problem, called $(PC)$.

$$(PC) \quad \min\{f_C(t_C) : 0 \leq t_C \leq T\}$$

**Assumption 2**: For each cluster $C$ the optimal cost of $(PC)$ is finite. Either $f_C(t)$ has a unique minimizer, or there is a $t^*, 0 \leq t^* < T$, such that all $t \in [t^*, T]$ minimize $f_C(t)$.

**Lemma 2** Suppose that for each $y$ there is a $t^* \in [0, T]$ such that $F_{D,y}(t)$ decreases strictly in $t$ for $0 \leq t \leq t^*$, and is equal to 0 for all $t, t^* < t \leq T$. Then Assumption 2 holds.

Proof: Clearly for each $n$ there is a $t^*$ such that $\eta_n(t)$ is strictly convex for $t \leq t^*$ and is linear for $t^* > t$. This property is inherited by $f_C(t)$.

The motivation behind Assumption 2 will be understood after we introduce floor and shell expansions. When $(PC)$ has a unique minimizer, let $t_C$ denote it. Otherwise let $t_C := T$, the largest of the minimizers of $(PC)$. Given $J$, a set of clusters that constitute a partition of $\{1, ..., N - 1\}$, let $C(n)$ be the cluster in $J$ containing the $n$th machine. Thus $\min(C(n)) \leq n \leq \max(C(n))$ for all $n$. Let $R(n)$ be the roots of a set of clusters that give rise to an optimal solution to $(P^n)$, where $(P^n)$ is $(P)$ restricted to machines $\{1, ..., n\}$. Our Cluster Algorithm is shown in Table I. Its validity and running time are stated in the following Theorem.

- Table I -

**Theorem 1** The Cluster Algorithm produces an optimal solution to $(P)$. It takes $O(N \cdot T^c)$ time, where $T^c$ is the time required to solve a problem of the form $(PC)$. If (1) holds then $f_C(t)$ can be evaluated in time that is constant in $|C|$, and $T^c = O(1)$.

As promised before, we now consider $(P_{j,k})$, a version of $(P)$ with a nonzero fixed cost $G_n$ for purchasing machine $n$, in which we must purchase machines $n \leq j$, and cannot purchase machines $n \geq k$. Recall that $t_n = T$ if machine $n$ is not purchased, and that the total costs are the sum of the fixed costs $G_n$ and the variable costs $f_n$. Thus, $(P_{j,k})$ is
given by

\[(P_{j,k}) \quad \min_{j \leq s < k} \left\{ \min \{ \sum_{n=1}^{s} f_n(t_n) : 0 \leq t_1 \leq \ldots \leq t_s \leq T \} + G(s) \right\} \]

(2)

where \( G(s) = \sum_{n=1}^{s} G_n + \sum_{n=s+1}^{N-1} f_n(T) \). Resolve ties in favor of larger values of \( s \) and \( t_n \). Note that the inner minimization of (2) can be done by running the Cluster Algorithm on \((P)\). We use the Fixed-Cost Cluster Algorithm (see Table II) to solve \((P_{j,k})\). Clearly, the Fixed Cost Cluster Algorithm solves \((P_{j,k})\) in \( O(N \cdot T_c) \) time.

- Table II -

Appendix A contains a theorem on the structure of the optimal clusters and lemmas on the sensitivity of the solution to \((P)\) as machine costs \( f_n(.) \) vary. These lemmas play key roles in justifying our algorithm for simultaneously optimizing machine purchases, floor space and shell expansions.

4 Shop Floor and Shell Expansions

We now integrate the costs of floor space and shell expansions into our analysis. Let \( \{F_k : k \in \mathcal{F}\} \) and \( \{S_l : l \in \mathcal{S}\} \) be the set of all possible floor space and shell space levels, respectively. We assume that machine capacities \( a_n \) match floor and shell space levels, i.e., \( a_{k-1} = F_k \) if \( k \in \mathcal{F} \) and \( a_{l-1} = S_l \) if \( l \in \mathcal{S} \). Thus \( t_k \) is the time at which \( F_k \) units of floor space will cease to be adequate. The Floor and Shell Expansion Problem is the problem of determining optimal machine purchase times, and optimal times and sizes for both floor and shell expansions. We will solve the Floor and Shell Expansion Problem as a shortest path problem in a network. The nodes and arcs of the Expansion Network are defined in Table III (also see Figure 2). We let \( v \) stand for a generic node in the Expansion Network.

- Table III -

- Figure 2 –

Let \( F_{j(0)} \) and \( S_{l(0)} \) be the initial shop floor space and shell space levels, respectively. We assume that floor and shell expansions are nested i.e., \( \mathcal{F} \) and \( \mathcal{S} \) satisfy \( \{0, l(0), N\} \subseteq \mathcal{S} \subseteq \mathcal{F} \subseteq \mathcal{N} := \{0, \ldots, N\} \). Clearly \( j(0) \in \mathcal{F} \), \( l(0) \in \mathcal{S} \) and \( 0 < j(0) \leq l(0) \). We include 0 and \( N \) in \( \mathcal{F} \) and \( \mathcal{S} \) as a notational convenience. If there are no existing facilities
for manufacturing then 1 is a dummy machine, \( f_k(t) = 0 \), and \( j(0) = l(0) = 1 \). If shell expansions are not part of the problem then \( l(0) = N \). We use the short hand notation \((0)\) for the starting node \((S,0,j(0),l(0))\).

Every node \( v \) in the Expansion Network, except for \((0)\) and \((N)\), has an associated floor space, or floor and shell space expansion cost. This cost is denoted by \( H(t;v) \) if incurred at time \( t \). It includes the cost of the capital required for expansion, plus periodic maintenance costs. Specifically, \( H(t_j;F,j,k,l) \) is the floor space expansion cost from floor space \( a_{j-1} \) to \( a_{k-1} \). Similarly, \( H(t_j;S,j,k,l) \) is the cost of the floor space expansion from \( a_{j-1} \) to \( a_{k-1} \), plus the cost of a simultaneous shell expansion from \( a_{j-1} \) to \( a_{l-1} \). These costs are incurred at \( t_j \), the availability time of the \( j \)th machine. We assume that \( H(t;v) \) is continuous, non-negative, non-increasing and convex. Thus, \( H(T;v) \) captures the fixed expansion cost. If an expansion associated with a node is not performed, then no cost is incurred. We set \( H(t;v) = \infty \) if \( t \) is less than the lead time required to implement the expansion associated with node \( v \). Since an expansion cost \( H(t;v) \) is related to three different machines \( j \), \( k \) and \( l \), it is not possible to model a floor or shell expansion as a single phony machine expansion.

We now formulate the Floor and Shell Expansion Problem. For every non-terminal node \( v \), we let \( v^1 \) be the first machine index of node \( v \), \( v^2 \) and \( v^3 \) are similarly defined. Namely, \((\cdot,j,k,l)^1 = j, (\cdot,j,k,l)^2 = k \) and \((\cdot,j,k,l)^3 = l \). A path \( \pi = [(0), \ldots, (N)] \) from \((0)\) to \((N)\) in the Expansion Network defines the sequence of floor space and shell expansions that are implemented. Let \( last(\pi) \) be the last non-terminal node in path \( \pi \). Then, the expansion associated with \( last(\pi) \) raises the floor space from \( F_{last(\pi)^1-1} \) to \( F_{last(\pi)^2-1} \) and may also raise the shell space. Let \( s \) represent the last machine purchased during the planning horizon, i.e., \( s = \min\{n : t_n < T\} \). Note that \( s \geq last(\pi)^2 \) is inconsistent with the definition of \( last(\pi) \). Without loss of generality, we can consider only \( s \) where \( s \geq last(\pi)^1 \); otherwise the last expansion is not needed. Thus, \( last(\pi)^1 \leq s < last(\pi)^2 \). The Floor and Shell Expansion Problem can be formulated as

\[
(\mathcal{E}) : \quad \min_{\{\pi\}} \min_{\{s: last(\pi)^1 \leq s < last(\pi)^2\}} \{\min_{0 \leq t_1 \leq t_2 \leq \cdots \leq t_s \leq T} \left[ \sum_{n=1}^{s} [f_n(t_n) + \sum_{v \in \pi} H(t_{v^1} : v)] + G(s) \right] \}
\]

Clearly the minimum cost is attained. Since \( \pi = [(0), \ldots, (N)] \) with \( s = 0 \) has cost \( G(0) < \infty \), the minimum cost is finite. Note that the inner minimization in \((\mathcal{E})\) is of the form \((\mathcal{P})\). Consequently it can be done using the Cluster Algorithm, and the lemmas of the Appendix A apply. The minimization over \( s \) can be done using a slight modification.
of the Fixed-Cost Cluster Algorithm. Ties are resolved by favoring large \(t_n\)'s and \(s\)'s.

We will solve the Floor and Shell Expansion Problem using a shortest path algorithm. The challenge is to allocate the costs incurred to the arcs in \(\pi\) in a manner that is appropriate for all of the paths that pass through a given arc. The key to accomplishing this is Theorem 4, which allows us to optimize the \(t_n\)'s for each cluster independently. We compute arc lengths by solving a series of problems of the form of (\(P\)). Let

\[
 f^*_n(t) = \inf \{ f_n(u) : u \leq t \}.
\]

Note that in spite of this modification, Assumption 2 remains valid. For every non-terminal node \(v\) and for every \(s, v^1 \leq s < v^2\) we obtain \((\mathcal{E}_v^s)\) from \((P)\) by replacing \(f_{v^1}(t)\) with \(f_{v^1}(t) + H(t_{v^1}; v)\), and replacing \(f_n(t)\) with \(f^*_n(t)\) for all \(n > s\). We define \((\mathcal{E}_v)\) to be \((\mathcal{E}_v^{s^2-1})\). Let \(t_n(\mathcal{E}_v)\) be the optimal value of \(t_n\) in \((\mathcal{E}_v)\), and let \(t_n(\mathcal{E}_v^s)\) be similarly defined. Let \(C_v(n)\) be the optimal cluster that contains machine \(n\) in the solution to \((\mathcal{E}_v)\), and let \(C_v^s(n)\) be similarly defined for \((\mathcal{E}_v^s)\).

We represent a generic, non-terminal arc by \([v_1, v_2]\). Table III depicts the conditions for the existence of an arc between two arbitrary nodes. It does not make sense to implement two different floor space expansions at the same time — it would be better to integrate them into a single expansion. We define an existing non-terminal arc \([v_1, v_2]\) to be legal if the expansion in \(v_1\) naturally precedes the expansion in \(v_2\), i.e., if \(t_{v_1}(\mathcal{E}_{v_1}) < t_{v_2}(\mathcal{E}_{v_2})\), or if \(v_1 = (0)\).

Conceptually, to obtain the length of a legal arc \([v_1, v_2]\) we solve \((\mathcal{E}_{v_1})\). We attach the value of \(t_n\) and its cost \(f_n(t_n)\) to \([v_1, v_2]\) if \(n\) falls in or after the cluster containing \(v_1^1\), and before the cluster that contains \(v_2^1\) in \((\mathcal{E}_{v_2})\). We also attach \(H(t_{v_1}; v)\) to \([v_1, v_2]\). The fact that availability times \(t_n\) can be computed independently for different clusters enables us to combine the \(t_n\)'s attached to different arcs in a path \(\pi\) and to assemble a feasible solution to the Floor and Shell Expansion Problem \((\mathcal{E})\).

Formally, from the solution to \((\mathcal{E}_v)\) we define \(c^{\leq n}(\mathcal{E}_v^s) = \sum_{k=1}^n f_k(t_k) + H(t_{v^1}; v)1_{\{v^1 \leq n\}}\) and \(r(\mathcal{E}_v^s) = \min(C_v^s(v^1))\), see Figure 3. Let \(c^{\leq n}(\mathcal{E}_v)\) and \(r(\mathcal{E}_v)\) be defined similarly, with \(r(\mathcal{E}_{(0)}) = 1\). The length of a legal arc \([v_1, v_2]\) is

\[
 \lambda_{v_1, v_2} = c^{\leq r(\mathcal{E}_{v_2})^{-1}(\mathcal{E}_{v_1})} - c^{\leq r(\mathcal{E}_{v_1})^{-1}(\mathcal{E}_{v_1})}.
\]

(4)
Thus \( \lambda_{v_1, v_2} \) is the total cost associated with the machines in \( \{ n : r(\mathcal{E}_{v_1}) \leq n < r(\mathcal{E}_{v_2}) \} \) in an optimal solution to \( (\mathcal{E}_{v_1}) \).

Conversely, we say that the arc \([v_1, v_2]\) *imputes* the values \( t_n = t_n(\mathcal{E}_{v_1}) \) for all \( n, r(\mathcal{E}_{v_1}) \leq n < r(\mathcal{E}_{v_2}) \) from \( (\mathcal{E}_{v_1}) \). The length of an illegal, non-terminal arc \([v_1, v_2]\) is \( \lambda_{v_1, v_2} = \infty \).

To get the length of the terminal arc \([v, N]\), we have to consider fixed costs for buying machines. Fixed costs effect which machines are purchased and which are not. However once that decision has been made their values have no effect on the availability times \( t_n \). Let \( q(\mathcal{E}_{v}^s) \) denote the optimal index of the last machine purchased, given that the index must be smaller than or equal to \( s \). Thus,

\[
q(\mathcal{E}_{v}^s) = \arg\min_{v_1 \leq m \leq s} [c^{\leq m}(\mathcal{E}_{v}^m) + G(m)] \quad \text{defined for } v_1 \leq s < v_2. 
\]  

(5)

As before, break ties in favor of large \( m \)'s.

If \( v_1 \leq s < s' < v_2 \) then \((\mathcal{E}_{v}^s)\) and \((\mathcal{E}_{v}^{s'})\) differ in that in \((\mathcal{E}_{v}^s)\), \( f_n^*(t) \) is used in place of \( f_n(t) \) for \( s < n \leq s' \). If we imagine a transition from the solution of \((\mathcal{E}_{v}^{s'})\) to the solution of \((\mathcal{E}_{v}^s)\), the changing costs push the availability times \( t_n, s < n \leq s' \) out to \( T \). Recall that if \( t_n = T \) then machine \( n \) is not purchased. According to Lemma 10 in Appendix A, the values of \( t_n \) for \( s \geq n \geq r(\mathcal{E}_{v}^{s'}) \) may increase, and \( C_v^{s'}(v_1) \) may split into smaller clusters. However \( t_n \) does not change for \( n < r(\mathcal{E}_{v}^{s'}) \). Consequently,

\[
\text{There is a cluster break at } (r(\mathcal{E}_{v}^{s'}) - 1, r(\mathcal{E}_{v}^{s'})) \text{ in } (\mathcal{E}_{v}^s), \text{ if } v_1 \leq s < s' < v_2, \text{ and } \\
c^{\leq n}(\mathcal{E}_{v}^s) = c^{\leq n}(\mathcal{E}_{v}^{s'}) \text{ for any } n, s, s' \text{ such that } v_1 \leq s < s' < v_2 \text{ and } n < r(\mathcal{E}_{v}^{s'}). 
\]  

(6)

(7)

By a cluster break at \((n-1, n)\) in a problem \((\mathcal{E})\), we mean to say that machines \( n-1 \) and \( n \) end up in different clusters in \((\mathcal{E})\).

To compute the length of \([v, N]\) we solve \((\mathcal{E}_{v}^s)\) for each \( s, v_1 \leq s < v_2 \) and set \( q = q(\mathcal{E}_{v}^{s-1}) = q(\mathcal{E}_{v}). \) We sum \( f_n(t_n) \) for all \( n \leq q \) that fall in or after the cluster that contains \( j \) in \((\mathcal{E}_{v})\), where \( t_n \) comes from \((\mathcal{E}_{v}^q)\). We add \( H(t_n; v) \) and the fixed costs associated with all machine purchases, and \( f_n(T) \) for all machines \( n \) not purchased. Using (7) with
\[ s = q \text{ and } s' = v^2 - 1, \text{ the length of } [v, N] \text{ is formally} \]
\[ \lambda_{v,N} = c^{s}(\mathcal{E}^a) - c^{s}(\mathcal{E})^{-1}(\mathcal{E}) + G(q), \text{ where } q = q(\mathcal{E}). \] \hspace{1cm} (8)

The terminal arc \([v, N]\) imputes the values \(t_n = t_n(\mathcal{E})\) from \((\mathcal{E})\) for \(r(\mathcal{E}) \leq n \leq q\), and \(t_n = T\) for \(q < n < N\).

We have now assigned lengths to all arcs in the Expansion Network of Figure 2. Note that all fixed costs associated with machine purchases are attached to terminal arcs. If \(G_n = 0\) for all \(n\), i.e., that there are no fixed costs for buying machines, we will subsequently show that a shortest path from \((0)\) to \((N)\) in the Expansion Network (Figure 2) determines the optimal solution to \((\mathcal{E})\).

When there are fixed costs for purchasing machines, we must enrich the Expansion Network with splitting arcs. If \([v_1, v_2]\) is an illegal arc, i.e., if \(t_{v_2}^{e_2}(\mathcal{E}) \leq t_{v_1}^{e_1}(\mathcal{E})\), then we attempt to create the splitting arc \([v_1, v_2, N]\). Traversing \([v_1, v_2, N]\) means deciding not to purchase the machines in \(\{s+1, s+2, .., N\} \cap C(v_2)\) for some \(s\), in order to to achieve \(t_{v_2}^{e_2}(\mathcal{E}) > t_{v_1}^{e_1}(\mathcal{E})\). Constraining \(t_n = T\) for \(n \in \{s+1, s+2, .., N\} \cap C(v_2)\) causes an increase in variable costs; hopefully the savings in fixed costs are large enough to compensate for the increase in variable costs. The following example illustrates the value of splitting arcs.

**Example:** Let \(T = 5, N = 5, \mathcal{F} = \{0, 2, 3, 5\},\) and \(\mathcal{S} = \{0, 5\}\). The Expansion Network appears in Figure 4. The variable machine acquisition costs are \(f_1(t_1) = t_1^2, f_2(t_2) = (t_2 - 2)^2, f_3(t_3) = (t_3 - 4)^2\) and \(f_4(t_4) = (t_4 + 6)^2\). Fixed machine acquisition costs are found in Table IV. Since \(\mathcal{S} = \{0, 5\}\) shell expansions are not part of the problem. The floor space expansion costs are \(H(t; F, 2, 3, 5) = 1, H(t; F, 3, 5, 5) = [\min(0, t - 4)]^2\) and \(H(t; F, 2, 5, 5) = 1 + [\min(0, t - 4)]^2\). Table V contains the solutions to the sub-problems that need to be solved. The arc \([F, 2, 3, 5], (F, 3, 5, 5)\] is illegal because \(t_2(\mathcal{E}_{(F,2,3,5)}) = 2 > 1 = t_3(\mathcal{E}_{(F,3,5,5)})\). The computation of the other arc lengths is summarized in Table VI.
The shortest path from \((0) = (S,0,2,5)\) to \((5)\) has length 123. However the optimal solution to this problem is \((t_1,\ldots,t_4) = (0,2,4,5)\) with capacity expansions corresponding to \((F,2,3,5)\) at \(t_2 = 2\) and \((F,3,5,5)\) at \(t_3 = 4\). The total cost is \(0^2 + (2 - 2)^2 + (4 - 4)^2 + (5 + 6)^2 + 1 + \{[4 - 4]^2\} = 122\). To capture this solution we create a splitting arc \([\{(F,2,3,5),(F,3,5,5)\},5]\) connecting nodes \((F,2,3,5)\) and \((5)\). Since \([\{(F,2,3,5),(F,3,5,5)\}]\) is illegal, we constrain \(t_4 = 5\) and consider \(\{E_{(F,3,5,5)}^3\}\) rather than \([E_{(F,3,5,5)}^3]\). The constraint \(t_4 = 5\) effectively splits the cluster \(C_{(F,3,5,5)}(3) = \{2,3,4\}\) with \(t_2 = t_3 = t_4 = 1\) into three single-machine clusters with availability times \(t_2 = 2, t_3 = 4, t_4 = 5\), taken from the solution to \([E_{(F,3,5,5)}^3]\). Since \(t_4\) has increased from 1 to 4, the illegality of \([\{(F,2,3,5),(F,3,5,5)\}]\) has been circumvented. As an added benefit, we avoid paying the large fixed cost associated with machine 4. The new splitting arc imputes \(t_2 = 2, t_3 = 4, t_4 = 5\) and has a cost of 122, leading us to the optimal solution.

Formally, arc \([v_1,v_2]\) is \(s-\text{legal}\) if \(t_{v_1}^1(E_{v_1}) < t_{v_2}^1(E_{v_2}^s)\). A legal arc \([v_1,v_2]\) is \((v_2^2 - 1)\)-legal.

**Lemma 3** \(t_{v_2}^1(E_{v_2}^s)\) is non-increasing in \(s\). Consequently \(\{s : [v_1,v_2] \text{ is } s-\text{legal}\} = \{s : v_2^1 \leq s \leq s_{v_1,v_2}\}\) where

\[
s_{v_1,v_2} = \max\{s : v_2^1 \leq s \text{ and } t_{v_1}^1(E_{v_1}) < t_{v_2}^1(E_{v_2}^s)\}.
\]

If \(s_{v_1,v_2}\) exists and \([v_1,v_2]\) is illegal then

\[
s_{v_1,v_2} < \max(C_{v_2}(v_2^1)). \tag{9}
\]

Proof: The first assertion holds by Lemma 10(c) in Appendix A. For \(s \geq \max(C_{v_2}(v_2^1))\), by Lemma 10(b) in Appendix A and the illegality of \([v_1,v_2]\), \(t_{v_2}^1(E_{v_2}^s) = t_{v_2}^1(E_{v_2}) \leq t_{v_1}^1(E_{v_1})\), so (9) holds. □

The lemma implies that \(s_{v_1,v_2}\) exists if and only if \(t_{v_1}^1(E_{v_1}) < t_{v_2}^1(E_{v_2}^s)\).

If \([v_1,v_2]\) is illegal and \(s_{v_1,v_2}\) exists we create the splitting arc \([v_1,v_2,N]\), connecting nodes \(v_1\) and \((N)\) (see Table VII). We get the length of \([v_1,v_2,N]\) by optimizing over the index of the last machine purchased. The optimal index is \(q(E_{v_2}^{s_{v_1,v_2}})\), given by (5). The computation of \(q(E_{v_2}^{s_{v_1,v_2}})\) is simplified by noting that the \(v_1^1\)th expansion costs affect \(q(E_{v_2}^{s_{v_1,v_2}})\) only through \(s_{v_1,v_2}\).
Let \( q = q(E_{v_1}^{s,v_2}) \) and \( r = r(E_{v_2}^{q}) \). We compute the length of \([v_1, v_2, N]\) in two parts. The splitting arc \([v_1, v_2, N]\) imputes \( t_n, r(E_{v_1}) \leq n < r \) from \((E_{v_1})\); the associated costs are \( \lambda_{v_1,v_2,N}^{1} \). In addition \([v_1, v_2, N]\) imputes \( t_n, r \leq n \leq q \) from \((E_{v_2})\), and \( t_n = T \) for \( q < n < N \); the associated costs are \( \lambda_{v_1,v_2,N}^{2} \). Thus

\[
\lambda_{v_1,v_2,N}^{1} = c^{\leq r-1}(E_{v_1}) - c^{\leq r}(E_{v_1})^{-1} - c^{\leq r-1}(E_{v_1}),
\]

\[
\lambda_{v_1,v_2,N}^{2} = c^{\leq q}(E_{v_2}) - c^{\leq r-1}(E_{v_2}) + G(q),
\]

\[
\lambda_{v_1,v_2,N} = \lambda_{v_1,v_2,N}^{1} + \lambda_{v_1,v_2,N}^{2}
\]

(10)

In section 5 we show that splitting arcs are not needed when the fixed costs for purchasing machines are zero.

- Table VII -

5 The Fix Four Expansions (FIFEX) Algorithm

In the previous section we defined the nodes and the arcs of the Expansion Network; see Figure 2. We create the Cost Network when we add splitting arcs to the Expansion Network, as described in the previous section. In this section we give an algorithm that generates the Cost Network from the Expansion Network, and computes the lengths of all arcs. A shortest path problem is solved on the Cost Network. A path from \((0)\) to \((N)\) imputes a complete solution \((t_n : 1 \leq n < N)\) to \((E)\). Thus finding \(t_n\) is trivial once the shortest path is specified. Table VIII lays out Fix Four Expansions algorithm (FIFEX) to solve \((E)\).

The Cost Network has \( O(|F|^2|S|) \) nodes and \( O(|F|^3|S|) \) arcs. Steps A and B of (FIFEX) require \( O(N) \) time for each node, or \( O(|F|^2|S|N) \) overall. Note that \( q(E_v^s) \) is easily obtained from \( q(E_v^{s-1}) \). The lists \( U \) and \( V \) both are \( O(|F|^2|S|N) \) long. Steps C and D_1 require constant time to locate the data associated with \((E_v)\) and \((E_v^q(E_v))\), hence they take \( O(|F|^2|S|) \) and \( O(|F|^3|S|) \) time overall, respectively. In step D_2, an \( O(|F|) \)-long list \( W(v) \) is generated for each node. Thus step D_2 takes \( O(|F|^3|S|) \) time, and step E requires \( O(|F|\log|F|) \) time for each node, or \( O(|F|^3|S|\log|F|) \) overall. When step F is reached \( W(v) \) is sorted, and \( t_{v,E}^{s}(E_v) \) is a nonincreasing function of \( s \). Therefore we can generate the \( s_{v_1,v_2} \) values for all elements of \( W(v_2) \) by making a single coordinated pass through two
sorted lists, in $O(N)$ operations for each node, or $O(|F|^2|S|N)$ for all nodes. Step G takes $O(1)$ per arc, or $O(|F|^3|S|)$ overall. We have proven the following theorem. Note that for problems of practical interest, $|F|$ is much smaller than $N$.

- Table VIII -

**Theorem 2** The overall run time for (FIFEX) is $O(|F|^2|S| \max\{N,|F|\log|F|\})$. FIFEX requires $O(|F|^2|S|N)$ memory space.

In two steps, we now prove that FIFEX solves ($\mathcal{E}$). First we establish that the cost of an optimal solution to ($\mathcal{E}$) is greater than or equal to the length of some path from (0) to (N) in the Cost Network. Second we argue that any path from (0) to (N) in the Cost Network imputes a feasible solution to ($\mathcal{E}$) whose cost is the length of that path. These steps correspond to the following two lemmas. The details of proofs of Lemmas 4, 5 and 6 are in Appendix B.

**Lemma 4** Given an optimal solution to ($\mathcal{E}$), we can identify a path from (0) to (N) whose length is less than or equal to the optimal cost.

Outline of Proof: Given an optimal solution to ($\mathcal{E}$) we use the capacity expansions performed to identify a path in the Cost Network. We allocate the costs incurred by the optimal solution to the arcs in the path, in the obvious manner. We then argue that the cost allocated to each arc in the path is greater than or equal to the arc length. This argument is based on the optimization problems used to define arc lengths.

**Lemma 5** Each path from (0) to (N) in the Cost Network imputes a feasible solution to ($\mathcal{E}$). The cost of the solution is the length of the path.

Outline of Proof: The path trivially imputes a complete solution to ($\mathcal{E}$). If the solution is feasible, then by construction its cost is equal to the length of the path. We use the properties of legal, terminal and splitting arcs to prove feasibility.

**Theorem 3** FIFEX computes an optimal solution to ($\mathcal{E}$).
Proof: A direct consequence of Lemmas 4 and 5. □

We end this section by considering the case in which there are no fixed costs for acquiring machines, i.e., $G_n = 0$ for all $n$. One situation in which this case is of interest is when demand will continue to grow strongly after the end of the time horizon. In that case all of the machines being modeled will eventually be purchased, it is just a matter of timing. We claim that splitting arcs are not required in this special case.

**Lemma 6** If $G_n = 0$ for all $n$ then there is an optimal path from (0) to (N) in the Cost Network that does not use splitting arcs.

Outline of Proof: We assume that an optimal path uses the splitting arc $[v_1, v_2, N]$. We perturb the imputed solution by setting $t_n = T - \epsilon$ for all $n \in C_{v_2}(v_1^2)$ satisfying $t_n = T$. This contradicts optimality.

As a result of this lemma, FIFEX can be streamlined when $G_n = 0$ for all $n$. We only solve $({\mathcal E}_e)$ in step $A$, and none of the steps $D_2, E, F, G$ is needed. Then the FIFEX Algorithm takes $O(|F|^2|S|N)$.

### 6 A Real Life Example

We obtained tool data from Sematech (SEmiconductor MAufacturing TECHnology) databases. Tool data includes purchase prices (in $), capacities (in number of wafers per month), area requirements (in m$^2$) and delivery lead times (in months). There are about 50 tool 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 there are no fixed costs for purchasing machines. Clean-room space requirements per tool range from 2 m$^2$ to 40 m$^2$. Initial fab capacity is 1844 wafers per month. We plan for capacity expansions starting 12 months from now and ending 72 months from now, so lead times shorter than or equal to 12 months are irrelevant. We plan to buy at most 57 machines in the 60 month planning horizon ($N = 58$).

Floor space can be expanded before putting in the 22nd, 30th and 40th machines ($F = \{0, 22, 30, 40, 58\}$). The
corresponding numbers for shell expansions are 22 and 40 ($S = \{0, 22, 40, 58\}$). Floor space and shell (construction) lead times are 20 and 30 months, respectively. Both types of construction costs are affine. Floor space costs have an intercept at $10$ M go up by $7.25$ M for each thousand-wafer-manufacturing space. Corresponding numbers are $23$ M and $2.85$ M for shell space.

The Cost Network has 12 nodes and 23 arcs and is depicted in Figure 5. There are 10 non-terminal nodes, 4 of type $(F, j, k, l)$ and 6 of type $(S, j, k, l)$.

– Figure 5 –

We recover the optimal capacity expansion schedule from the shortest path in Figure 5. In Figure 6 Fab capacity is graphed, along with the lower and upper bounds of the (trapezoidal) demand density. There are only 8 different values for the availability times $t_n$ although more than 8 machines are purchased. This is partly a result of the fixed cost for the facility expansion that takes place around 30th month. However the clustering of machine availability times is mostly driven by the economics of tool acquisition.

– Figure 6 –

7 Conclusion

We developed the polynomial time FIFEX algorithm to compute optimal machine capacity, shop floor and shell space expansion times. These capacity expansions have considerable lead times and involve investing large amounts of capital in the face of uncertainty. They are the primary irreversible decisions that have long term effects on competitiveness and profitability. Hence it is desirable to make reasonable decisions far in advance, when uncertainty in the demand is large. Capacity expansion plans must consider the risks that arise from uncertainty, which can only be built in via a stochastic demand model. Thus, we believe that our model is financially important and our assumptions are general enough for industrial applications. Yet the FIFEX algorithm efficiently solves this general model.

A good portion of the existing capacity expansion literature deals with single machine types. Multiple-machine type models are generally heuristics. FIFEX fills this gap providing an optimal solution to a multiple machine type
problem, under positive lead times and stochastic demands.

As far as we know, FIFEX is the only model in the literature optimizing expansions of nested limitations (shell and floor space) along with machine capacity expansions. In general, practitioners use hierarchical models where floor space expansions are at a higher level than machine capacity expansions. However, this approach is bound to create suboptimality which is overcome by FIFEX.

As a more technical point, FIFEX uses continuous availability times which allows for modelling the machine capacity expansion problem as an easily solvable nonlinear program instead of the more traditional approach of stochastic integer programming. Also note that Cluster Algorithm is efficient and FIFEX will inherit this efficiency. That is because, in practice the number of potential floor and shell space expansions ($|\mathcal{F}|$ and $|\mathcal{S}|$) will be small in comparison with $N$.

FIFEX makes several assumptions which may be violated in some practical situations. First FIFEX assumes known machine purchase, qualification and installation lead times while especially machine qualification times may be stochastic. Second related to lead times is machine capacities, our assumption of machine capacity being constant during the horizon may not hold. During a machine’s qualification, its productive capacity increases. Third FIFEX deals with a single product, there could be several products manufactured at the same facility. Relaxing each of these assumptions can provide a venue for future research.
Appendix A: Structural and Sensitivity Results for Optimal Clusters

In this appendix we will present the structure of optimal clusters and prove results on the sensitivity of the solution to $(P)$ as machine costs vary. These lemmas are used in studying floor and shell space expansions. First we state a useful property about the optimal availability time of two clusters when they are united:

- **Cluster Union Property**: For two nonempty disjoint clusters $C_1$ and $C_2$, if $t_{C_1} \le t_{C_2}$, then $t_{C_1} \le t_{C_1 \cup C_2} \le t_{C_2}$.

This property holds under our assumptions.

We can now state our theorem on optimal cluster structure. The proof can be found in [10].

**Theorem 4** Let $J$ be a partition of $\{1, \ldots, N - 1\}$ into clusters, that has the following properties.

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

(ii) (Dual feasibility): For all $C \in J$, if $C' := C \cap \{1, \ldots, n\}$ and $C'' := C \setminus C'$ are nonempty, then $t_{C'} > t_{C''}$.

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

An important consequence of Theorem 4 is that if we know where the breaks between clusters are, we can optimize each cluster separately by solving $(P_C)$. Cluster Union and Dual Feasibility properties imply that

$$t_{C''} \le t_C \le t_{C'}.$$  \hspace{1cm} (11)

Next we prove Lemmas about sensitivity.

**Lemma 7** If $C$ is an optimal cluster for a problem of the form of $(P)$ then

(i) $t_C \le t_{(\min(C), \ldots, k)}$ for all $k \ge \min(C)$,

(ii) $t_C \ge t_{(j, \ldots, \max(C))}$ for all $j \le \max(C)$.

Proof: We only prove (i); the proof of (ii) is similar. If $C = \{\min(C), \ldots, k\}$ the result is trivial, and if $C \supset \{\min(C), \ldots, k\}$ (Figure 7 Case 1) then equation (11) implies $t_C \le t_{(\min(C), \ldots, k)}$. If $C \subset \{\min(C), \ldots, k\}$ (Figure 7 Case 2), then $\{\min(C), \ldots, k\} = C_1 \cup C_2 \cup \cdots \cup C_{p-1} \cup \left(C_p \cap \{1, \ldots, k\}\right)$ for optimal clusters $C_i$, $1 \le i \le p$, $C = C_1$. By (11), $t_{C_i \cap \{1, \ldots, k\}} \ge t_{C_p}$. Primal feasibility implies that $t_C \le t_{C_i}$ for $1 \le i \le p$. By Cluster Union Property, (i) holds. $\square$
Proof: We prove (i) only. Assume by the way of contradiction that 
\[ i < j \]
a smallest index 
\[ t \]
before or after the change. Lemma 9 establishes that if \( i < j \) and \( i, j \) are in different clusters, then changing \( f_j \) might decrease \( t_i \) but cannot increase it.

**Lemma 8** Let \((Q)\) and \((Q')\) be problems of the form of \((P)\), identical except that either:

(i) For some \( j \), \( f_n \) and \( f'_n \) may differ for \( n \geq j \), or

(ii) For some \( j \), \( f_n \) and \( f'_n \) may differ for \( n \leq j \).

In Case (i) let \( i < j \); in Case (ii) let \( i > j \). Then at least one of the following happens:

(a) \( C(i) = C'(i) \)

(b) \( j \in C(i) \)

(c) \( j \in C'(i) \)

Proof: We prove (i) only. Assume by the way of contradiction that \( j \notin C(i) \cup C'(i) \) and \( C(i) \neq C'(i) \). Then there is a smallest index \( l, l \leq i \) such that \( C(l) \neq C'(l) \). Let \( C(l) = \{l, l+1, ..m\} \), \( C'(l) = \{l, l+1, ..m'\} \), and without loss of generality let \( m < m' \). Then 
\[ t_{C(l)} = t'_{C'(l)} \geq t_{C'(l) \cap C(l)} = t'_{C'(l) \cap C(l)} \geq t_{C(m+1)} \geq t_{C(l)} \]
establishing a contradiction. \( \square \)

**Lemma 9** Let \((Q)\) and \((Q')\) be problems of the form of \((P)\), identical except that \( f_n \) and \( f'_n \) may differ for \( n \geq j \). Let \( i < j \). If \( j \notin C(i) \) then 
\[ t'_{C'(l)} \leq t_{C(l)} \]

Proof: If \( j \notin C'(i) \) then Lemma 8 implies that \( C(i) = C'(i) \), so the result holds. Assume that \( j \in C'(i) \). Let \( k := max(C(i)) \) and \( \tilde{C} := C'(i) \cap \{1, ..., k\} \). Then \( i \leq k < j \), and 
\[ t'_{C'(i)} \leq t'_{\tilde{C}} = t_{\tilde{C}} \leq t_{C(i)} \]
\( \square \)

Let \( \frac{d}{dt} f_n(t) \) be the left-hand derivative of \( f_n(t) \). We write \( f_n \geq f'_n \) to indicate that \( \frac{d}{dt} f_n(t) \geq \frac{d}{dt} f'_n(t) \) whenever the derivatives exist. Lemma 10 states that increasing derivatives cannot decrease \( t_m \) for any \( m \), and that machines whose cost function derivatives increase shift towards machines with smaller indices.

**Lemma 10** Let \((Q)\) and \((Q')\) have the form of \((P)\), and be identical except that \( f_n \) and \( f'_n \) can differ for \( p \leq n \leq q \). Assume that \( f_n \geq f'_n \) for \( p \leq n \leq q \). Then
(a). \( \min(C(p)) \leq \min(C'(p)) \) and \( \max(C(q)) \leq \max(C'(q)) \).

(b). If either \( m < \min(C(p)) \) or \( m > \max(C'(q)) \) then \( C(m) = C'(m) \) and \( t_{C(m)} = t'_{C'(m)} \).

(c). \( t_{C(m)} \leq t'_{C'(m)} \) for all \( m \).

Proof: (a). Let \( l' = \min(C'(p)) \), \( k' = \max(C'(p)) \) and \( l = \min(C(p)) \). If \( l' < l \) then \( t_{C(p)} \geq t_{C(l-1)} \geq t_{C(l-1)} \geq t_{C(p)} \), a contradiction. The proof of \( \max(C(q)) \leq \max(C'(q)) \) is similar.

(b). In either case \( m \notin \cup_{p \leq n} \{ C(n) \cup C'(n) \} \). Lemma 8 with \( i = m \) establishes (b).

(c). Let \( C(m) = \{ j, ..., l \} \) and \( C'(m) = \{ j', ..., l' \} \). Then \( t_{C(m)} \leq t_{C'(j', ..., l')} \leq t'_{C'(j', ..., l')} \leq t'_{C'(m)} \). \( \square \)

Appendix B: The Proofs of Lemmas 4, 5 and 6

In this appendix we prove lemmas 4 and 5, which justify the FIFEX algorithm. We also prove Lemma 6. Before doing so we introduce some new notation and prove Lemma 11. We obtain the problem \((\mathcal{E}_{v_1,v_2}^*)\) from \((\mathcal{E}_{v_2}^*)\) by adding the expansion cost \( H(t_{v_1^1};v_1) \) to \( f_{v_1^1}(t_{v_1^1}) \). We denote \((\mathcal{E}_{v_1,v_2}^{*2-1})\) with \((\mathcal{E}_{v_1,v_2})\). Let \( C_{v_1,v_2}^*(n) \) be the optimal cluster that contains machine \( n \) in the solution to \((\mathcal{E}_{v_1,v_2}^*)\) and let \( t_n(\mathcal{E}_{v_1,v_2}^*) \) be the value of \( t_n \) in that solution. Note that \( t_{C_{v_2}^*(n)} = t_n(\mathcal{E}_{v_2}^*) \) and \( t_{C_{v_1,v_2}} = t_n(\mathcal{E}_{v_1,v_2}) \). For algorithmic discussions we used \( t_n(\mathcal{E}_{v_2}^*) \), but for proofs \( t_{C_{v_2}^*(n)} \) is often more convenient. Throughout this appendix, let \( i = v_1^1 \), \( j = v_2^1 = v_1^1 \) and \( k = v_2^2 \). The next lemma establishes relationships between optimal clusters of \((\mathcal{E}_{v_1})\), \((\mathcal{E}_{v_1,v_2}^*)\) and \((\mathcal{E}_{v_2}^*)\).

**Lemma 11** Suppose that the non-terminal arc \([v_1,v_2]\) is s-legal where \( j \leq s < k \). Then

(a). \( C_{v_2}^*(n) = C_{v_1,v_2}^*(n) \) if \( \max(C_{v_1}(i)) < n \).

(b). \( C_{v_1}(n) = C_{v_1,v_2}^*(n) \) if \( n < r(\mathcal{E}_{v_2}^*) \).

Proof: Note that in \((\mathcal{E}_{v_2}^*)\), \( f_n'(T) = 0 \) for all \( n > s \). By Dual Feasibility,

\[
\text{In } (\mathcal{E}_{v_2}^*), \text{ } n \text{ and } n + 1 \text{ are in different clusters, and } t_n = T, \text{ for all } n \geq s. \quad (12)
\]
We claim that \( C_{v_1}(i) = C^s_{v_1,v_2}(i) \). By (12) there is a cluster break at \((j-1, j)\) in \((E_{v_1})\), so \(i\) and \(j\) are in different clusters in \((E_{v_1})\). Also, by the \(s\)-legality of \([v_1, v_2]\) and recalling that \((E_{v_1-j+1}) = (E_{v_1})\),

\[
t_{C^s_{v_1,v_2}(j)}^{\text{lem.10(c)}} \geq t_{C^s_{v_2}(j)} > t_{C_{v_1}(i)} \geq t_{C^s_{v_1,v_2}(i)}.
\]

(13)

Thus, \(i\) and \(j\) are in different clusters in \((E_{v_1})\) and \((E^s_{v_1,v_2})\). By Lemma 8 our claim holds.

(a). By our claim, \(n > \max(C^s_{v_1,v_2}(i))\). Lemma 10(b) applies with \((\mathcal{Q}) = (E^s_{v_2})\), \((\mathcal{Q}') = (E^s_{v_1,v_2})\) and \(p = q = i\).

(b). \(r(E^s_{v_2}) = \min(C^s_{v_2}(j)) \overset{(a)}{=} \min(C^s_{v_1,v_2}(j))\), so \(n \notin C^s_{v_1,v_2}(j)\). By (12), \(n \notin C_{v_1}(j)\). Lemma 8 completes the proof. \(\square\)

**Lemma 4** Given an optimal solution to \((E)\), we can identify a path from \((0)\) to \((N)\) whose length is less than or equal to the optimal cost.

Proof: We identify the path \(\pi\) in the Expansion Network that corresponds to the shop floor and shell expansions in the optimal solution to \((E)\). Then we use the Cluster Algorithm to solve

\[
(E_{\pi}) : \quad \min_{\{s: \text{last}(\pi)^1 \leq s < \text{last}(\pi)^2\}} \{\min_{\{0 \leq t_1 < t_2 \leq \ldots < t_s \leq T\}} \left[ \sum_{n=1}^{s} [f_n(t_n) + \sum_{v \in \pi} H(t_v; v)] + G(s) \right] \}.
\]

As before, we favor large \(t_n\)'s and \(s\)'s when breaking ties. This gives us an optimal solution \((t^0_n)\) to \((E)\) whose clusters \(C(E_{\pi})(n)\) have the properties associated with the Cluster Algorithm. If the last two expansions correspond to \(v_1\) and \(v_2\), and if \(t_i(E_{v_1}) \geq t_j(E_{v_2})\), then we alter \(\pi\) by replacing the last 2 arcs with a splitting arc. At most one expansion happens at a given time in an optimal solution. Thus, if either the non-terminal arc \([v_1, v_2]\) or the splitting arc \([v_1, v_2, N]\) is in \(\pi\), then \(t^0_n < t^0_j\).

Let \(q^*\) be the last machine purchased, i.e. \(q^* = \max\{n : t^0_n < T\}\). Thus \(q^*\) and \(q^* + 1\) are in different clusters in \((E_{\pi})\), i.e.

\[
\min(C(E_{\pi})(q^* + 1)) = q^* + 1
\]

(14)

Then one of the following holds:
i. \( v_2 \) is a node in \( \pi \). That is followed by either a non-terminal arc or a splitting arc and \( q = k - 1 \).

ii. \([v_2, N]\) is a terminal arc in \( \pi \), \( q = q^* \) and \( j \leq q < k \).

iii. \([v_1, v_2, N]\) is a splitting arc in \( \pi \), \( q = q^* \) and \( j \leq q < k \).

Recall that \( i = v_1^1 \) is the machine index corresponding to the expansion \( v_1 \) which immediately precedes the expansion \( v_2 \) and \( j = v_2^1 \). If \( v_2 \) is the first expansion then \( C(\xi_{\pi})(i) = \{0\} \). We claim that

\[
C(\xi_{\pi})(n) = C(\xi_{\pi}')(n) \quad \text{and} \quad t_n^0 = t_n(\xi_{\pi}'_{v_2}) \quad \text{for all} \quad n, \max(C(\xi_{\pi})(i)) < n < \min(C(\xi_{\pi})(q + 1)).
\]

(15)

Note that (15) holds for \( n = j \). The first inequality in (15) is a consequence of \( t_i^0 < t_j^0 \). The second inequality follows from (14) for cases (ii) and (iii), and from \( t_j^0 < t_k^0 \) in case (i).

We prove (15) as follows. Create \( (\xi') \) from \( (\xi_{\pi}) \) by setting the costs of all expansions occurring before time \( t_j^0 \) to zero, i.e. \( H(t_n; \cdot) = 0 \) for all \( n \) such that \( n \leq i \). Solve \( (\xi') \) to obtain \( t_n' \). By Lemma 10(b), \( C(\xi_{\pi})(n) = C(\xi')(n) \) and \( t_n' = t_n^0 \) for all \( n > \max(C(\xi_{\pi})(i)) \). Now create \( \xi_{\pi}'_{v_2} \) from \( (\xi') \) by setting \( f_n(t) = f_n(t) \) for all \( n > q \). Lemma 10(b) establishes the claim.

The costs incurred by \( (\xi_{\pi}) \) are allocated to the arcs in \( \pi \) in the same manner that arc lengths are defined. We will prove the lemma by showing that for each arc in \( \pi \), the costs that are incurred by \( (\xi_{\pi}) \) and allocated to the arc are greater than or equal to the arc’s length.

We start by saying that for a non-terminal arc the allocated cost is equal to the arc length. If \([v_1, v_2]\) is a non-terminal arc in \( \pi \) then by (15), \( C(\xi_{\pi})(i) = C(\xi_{\pi})(i) \) and \( C(\xi_{\pi})(j) = C(\xi_{\pi})(j) \). Thus \( t_n^0 = t_n(\xi_{v_1}) \) for all \( n \) where \( r(\xi_{v_1}) \leq n < \min(C(\xi_{\pi})(j)) \). The equality of the arc length and the allocated cost is established for this case.

We now consider the last arc in \( \pi \), which is either a terminal or a splitting arc. Suppose that the terminal arc \([v_2, N]\) ends \( \pi \). By the argument preceding (3), \( j \leq q^* < k \). Since \( t_i^0 < t_j^0 \),

\[
\max(C(\xi_{\pi})(i)) < \min(C(\xi_{\pi})(j)) \quad \text{(15)} \Rightarrow \min(C(\xi_{\pi})(j)) = r(\xi_{v_2}).
\]

Thus (15) and (14) imply that \( t_n^0 = t_n(\xi_{v_2}) \) for all \( n, r(\xi_{v_2}) \leq n \leq q^* \). By (5), (7) and (8), the costs incurred by \( (\xi_{\pi}) \) and allocated to \([v_2, N]\) are greater than or equal to the arc length \( \lambda_{v_2, N} \).
Now suppose that the splitting arc \([v_1, v_2, N]\) ends \(\pi\). By the argument preceding (3), \(j \leq q^* < k\). Let \(r^* = r(E_{v_2}^{q^*})\). Applying (15) to \([v_1, v_2, N]\) we have \(r^* = \min(C_\pi'(j))\), and \(t_{n}^{0} = t_{n}(E_{v_2}^{q^*})\) for \(r^* \leq n \leq q^*\). Since \(v_1\) is in \(\pi\) we can apply (15) to it, so \(t_{n}^{0} = t_{n}(E_{v_1})\) for \(r(E_{v_1}) \leq n < r^*\).

Since \(t_{i}^{0} < t_{j}^{0}\), we have \(q^* \leq s_{v_1,v_2}\). We can define \(\lambda(q')\), \(j \leq q' \leq s_{v_1,v_2}\) using the expression for \(\lambda_{v_1,v_2,N}\) in (10), by replacing \(q\) with \(q'\) and \(r\) with \(r' = r(E_{v_2}^{q'})\). By construction, \(\lambda_{v_1,v_2,N} = \lambda(q(E_{v_2}^{v_1,v_2}))\). The costs that are incurred by \((E_{\pi})\) and allocated to \([v_1, v_2, N]\) are equal to \(\lambda(q^*)\). We will prove that \(\lambda(q^*) = \lambda_{v_1,v_2,N}\). It suffices to show that

\[
\lambda(q(E_{v_2}^{v_1,v_2})) = \min\{\lambda(q') : j \leq q' \leq s_{v_1,v_2}\}.
\]

Let \(\mu(m)\) represent the term in brackets in (5). It suffices to show that 

\[
\mu(q') - \lambda(q') = c^{\leq r' - 1}(E_{v_2}^{q'}) - c^{\leq r' - 1}(E_{v_1}) + c^{\leq r}(E_{v_1})^{-1}(E_{v_1}).
\]

Let \(m = \max(C_{v_1}(i))\). By Lemma 11, \(C_{v_2}(n) = C_{v_1,v_2}(n) = C_{v_1}(n)\) for \(m < n < r'\). Thus

\[
c^{\leq r' - 1}(E_{v_2}^{q'}) - c^{\leq r' - 1}(E_{v_1}) = c^{\leq m}(E_{v_2}^{q'}) - c^{\leq m}(E_{v_1}).
\]

By Lemma 10(a), \(r(E_{v_2}^{q'})\) is non-increasing in \(q\). Since \(j \leq q' < k\) and \(m < r(E_{v_2}^{q'})\), we can use (7) to get,

\[
c^{\leq m}(E_{v_2}^{q'}) = c^{\leq m}(E_{v_2}^{q'}).\]

Combining these equations we obtain

\[
\mu(q') - \lambda(q') = c^{\leq m}(E_{v_2}^{q'}) - c^{\leq m}(E_{v_1}) + c^{\leq r}(E_{v_1})^{-1}(E_{v_1}),
\]

which is independent of \(q'\). \(\Box\)

**Lemma 5** Each path from \((0)\) to \((N)\) in the Cost Network imputes a feasible solution to \((E)\). The cost of the solution is the length of the path.

**Proof:** Each path from \((0)\) to \((N)\) consists entirely of non-terminal Legal Arcs with the exception of the last arc, which is either a terminal arc or a splitting arc. Reviewing the manner in which arc lengths are defined and availability
times \(t_n\) are imputed from arcs, two facts are apparent. First, if the imputed solution is feasible for \((\mathcal{E})\), meaning that \(0 \leq t_{n-1} \leq t_n \leq T\) for all \(n\), then its cost is equal to the length of the path. Second, to show feasibility it suffices to show \(t_{n-1} \leq t_n\) when \(n-1\) and \(n\) are imputed from different optimization problems. Specifically, it suffices to consider \(n = r(\mathcal{E}^*_{v_2})\) where either \([v_1, v_2]\) is a legal non-terminal arc and \(s = k-1\), or where \([v_1, v_2, N]\) is an \(s\)-legal splitting arc.

\[
\begin{align*}
t_{r(\mathcal{E}^*_{v_2})-1} & \overset{\text{def}}{=} t_{C_{v_1}(r(\mathcal{E}^*_{v_2})-1)} = t_{C_{v_1,v_2}^*(r(\mathcal{E}^*_{v_2})-1)} \overset{\text{Thm.11(b)}}{=} t_{C_{v_1,v_2}(j)} \overset{\text{Thm.4(i)}}{\leq} t_{C_{v_1,v_2}(j)} \overset{\text{Thm.11(a)}}{=} t_{r(\mathcal{E}^*_{v_2})}. \quad \Box
\end{align*}
\]

\textbf{Lemma 6} If \(G_n = 0\) for all \(n\) then there is an optimal path from \((0)\) to \((N)\) in the Cost Network that does not use splitting arcs.

Proof: We assume that all of the fixed costs for floor space and shell expansions are strictly positive, perturbing then if necessary. Assume that an optimal path from \((0)\) to \((N)\) in the Cost Network uses the splitting arc \([v_1, v_2, N]\), and let \(s = s_{v_1,v_2}\). Then in the notation of (10), \(q = q(\mathcal{E}^*_{v_2})\). The availability times imputed by \([v_1, v_2, N]\) include the following: if \(r(\mathcal{E}^*_{v_2}) \leq n \leq s\) then \(t_n = t_{n}(\mathcal{E}^*_{v_2})\), and if \(s < n < N\) then \(t_n = T\). Let \(s' = \max\{n : t_n < T\} \leq s\). If \(s' < j\) then we can save \(H(T; v_2) > 0\) by deleting the expansion attached to machine \(j\), contradicting optimality. If \(s' \geq j\) then by (9), \(s' < m\) where \(m = \max(C_{v_2})\). Set \(C' = \{n : r(\mathcal{E}_{v_2}) \leq n \leq s'\}\) and \(C'' = \{n : s' < n \leq m\}\), so that \(C_{v_2}(j) = C' \cup C''\).

\[
\begin{align*}
t_{C''} & \overset{\text{(11)}}{\leq} t_{C_{v_2}(j)} \overset{\text{[v_1,v_2] illegal}}{\leq} t_{C_{v_1}(i)} \overset{\text{[v_1,v_2] s-legal}}{<} t_{C_{v_2}'(j)} \leq T.
\end{align*}
\]

By the definition of \(t_{C''}\) and the convexity of \(f_{C''}(\_): f_{C''}(T) > 0\) and we can reduce costs by setting \(t_n = T - \epsilon\) for all \(n \in C''\). \(\Box\)

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

25
References


Figure 1: Production capacity versus a realization of wafer demand.
Figure 2: Expansion Network for $F = \{0, 5, 8, 14, 20\}$, $S = \{0, 5, 8, 20\}$, $j(0) = l(0) = 5$. Terminal arcs are omitted.
Figure 3: The definition of $r(.)$. $i$ is a machine index corresponding to an expansion before $v$. 
Figure 4: The Expansion Network
Figure 5: Cost Network for $F = \{0, 22, 30, 40, 58\} \text{ and } S = \{0, 22, 40, 58\}$. Arc costs are in $\text{\$M}$. Solid line is the shortest path.
Figure 6: Solid lines are upper and lower bounds for demand density. • signs indicate the fab capacity.
Case 1: $C \supset \{\text{min}(C), \ldots, k\}$

Case 2: $C \supset \{\text{min}(C), \ldots, k\}$

Figure 7: Two cases of $C$ and $\{\text{min}(C), \ldots, k\}$ relative to each other
• INITIALIZE: $R(1) := \{1\}, C(1) = \{1\}, n := 2$

While $n < N$ do

• SOLVE $(P^n)$:

  • $C := \{n\}, R(n) := R(n - 1) \cup \{n\}, GraftComplete := false$

  While $\text{min}(C) > 1$ and $GraftComplete = false$ do

    • $k := \text{min}(C) - 1$

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

      • GRAFT: $R(n) := R(n) \setminus \text{min}(C)$, $C := C \cup C(k)$

    else

      • $GraftComplete := true$

  endwhile

  • $n := n + 1$

endwhile

Table I: Cluster Algorithm
- Use the Cluster Algorithm to compute the optimal costs for all $(P_{s,s} : j \leq s < k)$.
- Find $s^*$, the value of $s$ that minimizes (2).
- Use the Cluster Algorithm to compute the values of $t_n$ that solve $(P_{s^*,s^*})$.

Table II: Fixed-Cost Cluster Algorithm
NODES:

- Type \((F, j, k, l)\).

  Exists for all \(0 < j < k \leq l\) such that \(j, k \in \mathcal{F}, l \in \mathcal{S}\), and either \(l = l(0)\) or \(j > l(0)\).

  Indicates that before time \(t_j\) we installed floor space \(F_j\) and shell space \(S_l\). At \(t_j\) we will expand the floor space to \(F_k\).

- Type \((S, j, k, l)\).

  Exists for all \(0 < j < k \leq l\) such that \(k \in \mathcal{F}, j, l \in \mathcal{S}\). Also exists for \(j = 0, j(0) = k, l(0) = l\).

  For \(j > 0\), indicates that before \(t_j\) we installed floor space \(F_j\) and shell space \(S_j = F_j\). At \(t_j\) we will expand the floor space to \(F_k\) and the shell space to \(S_l\). For \(j = 0\), indicates the initial floor space and shell levels.

- The Terminal Node \((N)\).

ARCS:

- Non-Terminal Arcs: Each of the following arcs exists for all values of \(i, j, k, l\) such that the relevant, non-terminal nodes exist.

  - \([v_1, v_2]\), from \(v_1 = (F, i, j, l)\) to \(v_2 = (F, j, k, l)\).

  - \([v_1, v_2]\), from \(v_1 = (F, i, j, j)\) to \(v_2 = (S, j, k, l)\).

  - \([v_1, v_2]\), from \(v_1 = (S, i, j, l)\) to \(v_2 = (F, j, k, l)\).

  - \([v_1, v_2]\), from \(v_1 = (S, i, j, j)\) to \(v_2 = (S, j, k, l)\).

- Terminal Arcs: From each non-terminal node to node \((N)\).

  - Terminal Type \([v, N]\), from \(v = (F, j, k, l)\) to \((N)\).

  - Terminal Type \([v, N]\), from \(v = (S, j, k, l)\) to \((N)\).

Table III: Nodes and arcs of the Expansion Network
\[
G_n = 5^2 + (5 - 2)^2 + (5 - 4)^2 + (5 + 6)^2 = 156 \\
G(n) = 0 + (5 - 2)^2 + (5 - 4)^2 + (5 + 6)^2 = 131 \\
G(n) = 0 + 0 + (5 - 4)^2 + (5 + 6)^2 = 122 \\
G(n) = 0 + 0 + 0 + (5 + 6)^2 = 121 \\
G(n) = 0 + 0 + 0 + 100 = 100
\]

Table IV: Machine Purchase Costs
Table V: Solutions to \((E_v^*\).  

<p>| | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>((S, 0, 2, 5))</td>
<td>0</td>
<td>(5, 5, 5, 5)</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>0</td>
<td>0+156=156</td>
<td>0</td>
</tr>
<tr>
<td>((S, 0, 2, 5)^*)</td>
<td>1</td>
<td>(0, 5, 5, 5)</td>
<td>1</td>
<td>-</td>
<td>0^2=0</td>
<td>0</td>
<td>0+131=131</td>
<td>1</td>
</tr>
<tr>
<td>((F, 2, 3, 5)^*)</td>
<td>2</td>
<td>(0, 2, 5, 5)</td>
<td>2</td>
<td>0</td>
<td>0^2+(2-2)^2+1=1</td>
<td>1+122=123</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>((F, 2, 5, 5))</td>
<td>2</td>
<td>(0, 3, 5, 5)</td>
<td>2</td>
<td>0</td>
<td>0^2+(3-2)^2+{1+[3-4]^{2}} = 3</td>
<td>3+122=125</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>((F, 2, 5, 5))</td>
<td>3</td>
<td>(0, 3, 4, 5)</td>
<td>2</td>
<td>0</td>
<td>0^2+(3-2)^2+(0-0)^2+{1+[3-4]^{2}} = 3</td>
<td>3+121=124</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>((F, 2, 5, 5)^*)</td>
<td>4</td>
<td>(0, 1, 1, 1)</td>
<td>2</td>
<td>0</td>
<td>0^2+(1-2)^2+(1-4)^2+(1+6)^2+{1+[1-4]^{2}}=69</td>
<td>69+100=169</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>((F, 3, 5, 5))</td>
<td>3</td>
<td>(0, 2, 4, 5)</td>
<td>3</td>
<td>0</td>
<td>0^2+(2-2)^2+(4-4)^2+{4-4}^{2}=0</td>
<td>0+121=121</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>((F, 3, 5, 5)^*)</td>
<td>4</td>
<td>(0, 1, 1, 1)</td>
<td>2</td>
<td>0</td>
<td>0^2+(1-2)^2+(1-4)^2+(1+6)^2+{1-4}^{2}=68</td>
<td>68+100=168</td>
<td>3</td>
<td>0</td>
</tr>
</tbody>
</table>

Table V: Solutions to \((E_v^*\).  

\(c^1 = c^{r(E_v)}-1(E_v)\),  
\(c^2 = c^{q*(E_v^*)}\).  

"*" means \((E_v^*) = (E_v^{v-1})\).
\[
\begin{array}{|c|c|c|}
\hline
\text{Arc} & \text{Length} & \text{Imputed } t_n\text{'s} \\
\hline
[(S, 0, 2, 5), (F, 2, 3, 5)] & 0 - 0 = 0 & t_1 = 0 \\
\hline
[(S, 0, 2, 5), (F, 2, 5, 5)] & 0 - 0 = 0 & t_1 = 0 \\
\hline
[(F, 2, 3, 5), (F, 3, 5, 5)] & \infty \text{ (illegal arc)} & - \\
\hline
[(S, 0, 2, 5), (5)] & 0 - 0 + G(1) = 131 & t_1 = 0, t_2 = t_3 = t_4 = 5 \\
\hline
[(F, 2, 3, 5), (5)] & 1 - 0 + G(2) = 123 & t_2 = 2, t_3 = t_4 = 5 \\
\hline
[(F, 2, 5, 5), (5)] & 3 - 0 + G(3) = 124 & t_2 = 3, t_3 = 4, t_4 = 5 \\
\hline
[(F, 3, 5, 5), (5)] & 0 - 0 + G(3) = 121 & t_2 = 2, t_3 = 4, t_4 = 5 \\
\hline
\end{array}
\]

Table VI: Computation of Arc Lengths
SPLITTING ARCS:

For each illegal arc $[v_1, v_2]$ such that $s_{v_1,v_2}$ exists, there is a splitting arc from $v_1$ to $(N)$.

- Splitting Arc $[v_1, v_2, N]$ where $v_1 = (F, i, j, l)$ and $v_2 = (F, j, k, l)$.
- Splitting Arc $[v_1, v_2, N]$ where $v_1 = (F, i, j, j)$ and $v_2 = (S, j, k, l)$.
- Splitting Arc $[v_1, v_2, N]$ where $v_1 = (S, j, k, l)$ and $v_2 = (F, j, k, l)$.
- Splitting Arc $[v_1, v_2, N]$ where $v_1 = (S, j, j)$ and $v_2 = (S, j, k, l)$.

Table VII: Splitting Arcs of the Cost Network
• Initialize the Cost Network to the Expansion Network.
• Compute and record \( G(s) = \sum_{n \leq s} G_n + \sum_{n > s} f_n(T), 0 \leq s < N. \)
• For each node \( v_1 \) do the following, in decreasing order of \( v_1^1 \):
  • Set \( W(v_1) = \emptyset. \)

A: Solve \((E_{v_1}^s)\) for all \( v_1^1 \leq s < v_1^2 \) in increasing order of \( s \) using the Cluster Algorithm.

B: Record \( r(E_{v_1}^s), c^{\leq s}(E_{v_1}^s), c^{\leq r(q(E_{v_1}^q)) - 1}(E_{v_1}^s), q(E_{v_1}^s) \) and \( t_r(E_{v_1}^s) \) in \( U \) and \( c^{\leq n}(E_{v_1}) \) for \( 1 \leq n < v_1^2 \) in \( V. \)

C: Set the length of the terminal arc \([v_1,N] \), \( \lambda_{v_1,N} = c^{\leq q(E_{v_1}^q)}(E_{v_1}^q) - c^{\leq r(q(E_{v_1}^q)) - 1}(E_{v_1}) + G(q(E_{v_1}^q)) \).

• For all \( v_2 \) such that arc \([v_1,v_2] \) exists do D.

D: • If \( t_1(E_{v_1}) < t_2(E_{v_2}) \) then

  D1: Set the length of the non-terminal arc \([v_1,v_2] \), \( \lambda_{v_1,v_2} = c^{\leq r(E_{v_2}) - 1}(E_{v_1}) - c^{\leq r(E_{v_1}) - 1}(E_{v_1}). \)

  Else \( (\) the non-terminal arc \([v_1,v_2] \) is illegal).

  D2: Set \( \lambda_{v_1,v_2} = \infty \) and append \(((v_1), t_1(E_{v_1}))\) to \( W(v_2). \)

• Proceed to the next node \( v_1 \).

• For each node \( v_2 \) do the following:

E: Sort \( W(v_2) \) in the order of decreasing \( t_1(E_{v_1}) \).

• For all \(((v_1), t_1(E_{v_1})) \in W(v_2)\),
  • If \( t_1(E_{v_1}) < t_2(E_{v_2}) \) (i.e., \([v_1,v_2] \) is s-legal for some s) then

  F: Set \( s_{v_1,v_2} = \max \{ s : v_1^1 \leq s < v_2^2 \text{ and } t_1(E_{v_1}) < t_2(E_{v_2}) \}, q = q(E_{v_2}^{s_{v_1,v_2}}), r = r(E_{v_2}^q). \)

  G: Add the splitting arc \([v_1,v_2,N] \) to the Cost Network and set its length:

  \[ \lambda_{v_1,v_2,N} = c^{\leq r(E_{v_1}) - 1}(E_{v_1}) - c^{\leq r(q(E_{v_2}^q)) - 1}(E_{v_1}) + c^{\leq q(E_{v_2}^q)}(E_{v_1}) - c^{\leq r(E_{v_2}^q)}(E_{v_1}) + G(q). \]

  • Proceed to the next entry in \( W(v_2). \)

  • Proceed to the next node.

• Solve \( (E) \) by finding the shortest path from \((0)\) to \((N)\) in the Cost Network.

Table VIII: The FIFEX Algorithm

43