Minimum concave cost flows in capacitated grid networks

Shabbir Ahmed, Qie He, Shi Li, George L. Nemhauser

Abstract

We study the minimum concave cost flow problem over a two-dimensional grid network (CFG), where one dimension represents time periods and the other dimension represents echelons. The concave function over each arc is given by a value oracle. We give a characterization of the computational complexity of CFG based on the grid size (T periods and L echelons), the distribution of sources and sinks over the grid, and arc capacity values. For the capacitated case, we give a polynomial-time algorithm with fixed L, one echelon of sources and one echelon of sinks, and O(1) different capacity values, and a polynomial-time algorithm with two echelons, O(1) capacity values on arcs connecting echelons and arbitrary capacity on other arcs; These are likely the most general polynomially solvable cases since we show the problem becomes NP-hard if any conditions on the parameters is relaxed. For the uncapacitated case, we give a polynomial-time algorithm with fixed L and one echelon of sources (or sinks); we show that the problem is NP-hard if L is an input parameter or there are two echelons of sources and two echelons of sinks. Our algorithms and hardness results generalize complexity results for many variants of the lot-sizing problem, and answer several open questions on serial supply chains.

1 Introduction

We study the minimum concave cost flow problem over a two-dimensional grid network (CFG). Let \([n] = \{1, 2, \ldots, n\}\) for any \(n \in \mathbb{N}\). A grid network \(N = (V, A, b, U)\) is a directed acyclic graph with the node set \(V = \{v_{l,t} | l \in [L], t \in [T]\}\), the arc set

\[
A = \{(v_{l,t}, v_{l,t+1}) | l \in [L], t \in [T - 1]\} \cup \{(v_{l,t}, v_{l+1,t}) | t \in [T], l \in [L - 1]\},
\]

the supply function \(b : V \rightarrow \mathbb{R}\) which determines whether each node is a source \((b(v) > 0)\), a sink \((b(v) < 0)\) or a transshipment node \((b(v) = 0)\), and the vector \(U = (U_a) \in \mathbb{R}^{|A|}\) with \(U_a\) being the capacity for arc \(a\), as shown in Figure 1. We

---

*H. Milton Stewart School of Industrial & Systems Engineering, Georgia Institute of Technology, Atlanta GA 30332, USA. Email: sahmed@isye.gatech.edu.
†Department of Industrial & Systems Engineering, University of Minnesota, Minneapolis MN 55455, USA. Email: qhe@umn.edu.
‡Toyota Technological Institute at Chicago, Chicago IL 60637, USA. Email: shili@ttic.edu.
§H. Milton Stewart School of Industrial & Systems Engineering, Georgia Institute of Technology, Atlanta GA 30332, USA. Email: gnemhaus@isye.gatech.edu.
refer to the two subscripts \( l \) and \( t \) as the indices of \textit{echelon} and \textit{period}, respectively, so the grid network has \( L \) echelons and \( T \) periods. Define arcs \((v_{l,t}, v_{l,t+1}), (v_{l,t+1}, v_{l,t}), (v_{l,t}, v_{l+1,t})\) and \((v_{l+1,t}, v_{l,t})\) to be \textit{forward arc}, \textit{backward arc}, \textit{downward arc} and \textit{upward arc} respectively for any suitable \( l \) and \( t \). For now we assume that the grid network has only forward and downward arcs. Given a grid network \( \mathcal{N} = (V, A, b, U) \), CFG is to find a flow \( x \in \mathbb{R}^{|A|} \) to minimize

\[
\sum_{a \in A} c_a(x_a) \quad \text{s.t.} \quad \sum_{a \in \delta^+(v)} x_a - \sum_{a \in \delta^-(v)} x_a = b(v), \quad \forall v \in V, \quad 0 \leq x_a \leq U_a, \quad \forall a \in A,
\]

(1)

where \( c_a \) is the cost function for arc \( a \), and \( \delta^+(v) \) and \( \delta^-(v) \) are the set of outgoing and incoming arcs at node \( v \), respectively. We assume that the cost function \( c_a \) is a general nonnegative concave function represented by a value oracle for each \( a \in A \). Since the feasibility of CFG can be checked by solving an equivalent maximum flow problem in polynomial time, we assume that the problem is feasible in the rest of the paper. CFG was studied in [12], and is shown to be polynomially solvable in \( T \) and the number of queries of the value oracle for the uncapacitated case with fixed \( L \), one echelon of sources and two echelons of sinks and the uncapacitated case with fixed \( L \) and a single source. In this paper, we give complexity results for both capacitated and uncapacitated CFG with a more general distribution of sources and sinks. The main results are summarized in Tables 1 and 2 with new results highlighted.

\begin{table}[h]
\centering
\caption{Complexity of CFG with \( O(1) \) capacity values}
\begin{tabular}{|c|c|c|}
\hline
 & One echelons of sources and & Two echelons of sources \\
\hline
\text{Fixed} \( L \) & \text{P (new)} & \text{NP-hard (new)} \\
\hline
\text{General} \( L \) & ? & \text{NP-hard (new)} \\
\hline
\end{tabular}
\end{table}

All complexity results remain the same if the words “sources” and “sinks” in the tables are switched. We only consider \( O(1) \) capacity values in Table 1 since CFG with
Table 2: Complexity of uncapacitated CFG

<table>
<thead>
<tr>
<th></th>
<th>One echelon of sources and one echelon of sinks</th>
<th>One echelon of sources and multiple echelons of sinks</th>
<th>Two echelons of sources and two echelons of sinks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed $L$</td>
<td>P</td>
<td>P (new)</td>
<td>NP-hard (new)</td>
</tr>
<tr>
<td>General $L$</td>
<td>P</td>
<td>NP-hard (new)</td>
<td>NP-hard (new)</td>
</tr>
</tbody>
</table>

general capacities is NP-hard even with a single source and $L = 2$ (a reduction from the capacitated lot-sizing problem). As a complement, we do find a polynomial solvable special case with general capacities: the two-echelon CFG with general capacities on forward arcs and $O(1)$ capacity values on downward arcs. The complexity of CFG with general $L$, one echelon of sources and one echelon of sinks, and $O(1)$ capacity values is unknown.

The motivation to study CFG stems from applications in production planning, supply chain management, maritime transportation, and reverse logistics. One important example of CFG is the deterministic uncapacitated lot-sizing problem (ULS), which serves as a building block for the theory of production planning and inventory control. With fixed-charge type production cost and linear holding cost, ULS can be formulated as a two-echelon CFG with a single source at $v_{1,1}$. The initial study of ULS by Wagner and Whitin in the 1950s [22] has elicited much interest in studying ULS variants under more practical settings, like allowing backlogging or return of products, more general cost structures for production or inventory holding, multiple production steps, demands at intermediate echelons, capacity on production or inventory, among others. All these variants can be modeled as CFG by adjusting the network parameters or adding backward and upward arcs accordingly. The common feature is that the underlying network is a grid with a single source and the arc costs are concave. The interested reader is referred to [12] for more applications of the uncapacitated CFG.

Recently, there have been increasing interests in optimizing production, inventory holding, and distribution decisions simultaneously in a capacitated serial supply chain. Kaminsky and Simchi-Levi [14] studied a two-stage supply chain management problem with constant capacity at the production stage, and gave a polynomial time algorithm when the production cost is linear. Van Hoesel et al. [20] extended the model to a multi-echelon serial supply chain with constant capacity at the production stage and general concave costs for production and inventory holding, and gave a polynomial-time dynamic programming (DP) algorithm when the number of echelons is constant. Hwang et al. [13] considered the same problem as in [20] and gave a DP algorithm in time polynomial in both the number of echelons and the number of periods. These problems can be seen as special cases of CFG. To the best of our knowledge, so far capacity has only been considered at the production stage, and the capacity over each capacitated arc is the same in order to guarantee polynomial solvability. However in practice, capacities arise naturally at other stages of the supply chain, like the stages between the supplier and the manufacturer, the manufacturer and warehouses, the warehouses and retailers, or other stages of the productions for a serial production line. Previous mentioned algorithms exploited heavily the structure of the optimal solutions.
when capacities only appear at the first (production) stage. It is unlikely that these algorithms could be easily modified for problems with more general capacity structure. In fact, an open question raised in [14] is whether the two-stage problem can be solved under more general capacity assumptions. In [20], it was also mentioned that a research direction is to study serial supply chains in the presence of capacities at other levels in the chain.

In this paper, we give a polynomial-time algorithm for the multi-echelon serial supply chain problem with constant capacity at each echelon. We indeed prove a stronger result. On the one hand, we show that CFG is polynomial solvable when there are one echelon of sources, one echelon of sinks and \( O(1) \) capacity values (without the need to specifying the locations of capacitated arcs). Since the problems in [14] and [20] are specials cases of CFG with a single source, a single capacity value and capacitated arcs at the first echelon, our result generalizes these complexity results, and answers the open questions raised therein. On the other hand, we show that a three-echelon serial supply chain problem is NP-hard with intermediate demands and a single capacity value, leaving little hope for efficient algorithms for capacitated supply chain problem when intermediate demands are allowed. Besides the capacitated case, we also give a polynomial-time algorithm for the uncapacitated CFG when the number of echelon \( L \) is fixed and there is one echelon of sources (or sinks), therefore generalizing the results in Zangwill [23], Zhang et al. [24] and He et al. [12]. We answer a question in [12] by showing that the conditions (fixed \( L \) and one echelon of sources) are both crucial to the polynomial solvability of the problem.

We now survey some related results in the literature. Since the minimum concave cost flow problem (MCCFP) is NP-hard in general, various approximation algorithms and exact algorithms based on DP or branch-and-bound have been developed for specific network topology or cost functions (see [10] and [9] for a survey of recent results). Polynomial cases of MCCFP include a single-source problem with a single nonlinear arc cost [11], the flow problem with a fixed number of sources and nonlinear arc costs [16], a production-transportation network flow problem where the concave cost function is defined on only a constant number of variables [17], the pure remanufacturing problem [13] which can be seen as a two-echelon CFG with \( T \) sources and \( T \) sinks, and many variants of lot-sizing problems which we will cover shortly. There are not many results on MCCFP over grid networks. Erickson et al. [6] derived a DP algorithm for general MCCFP which runs in polynomial time if the graph is planar and all sources and sinks lie on a constant number of faces. The grid network is a planar graph, but the algorithm runs in exponential time in general if there are intermediate demands or arc capacities in the grid. Recently, it has been shown in [12] that uncapacitated CFG with one echelon of sources and two echelons of sinks can be solved in polynomial time. The literature on lot-sizing is abundant (see Pochet and Wolsey [15] for a detailed review). To be concise, we concentrate on problems that can be formulated as a CFG. For the uncapacitated case, ULS was first solved in \( O(T^2) \) by Wagner and Whitin [22], and the complexity was improved to \( O(T \ln T) \) later [1, 7, 21]; the multi-echelon ULS with demands at the last echelon was solved by Zangwill [23] with an \( O(LT^4) \)-time DP algorithm; two-echelon ULS with intermediate demands was recently studied in [24] for fixed-charge production costs and a new class of valid inequalities were derived. The capacitated lot-sizing problem is shown to be NP-hard [4], but it can be solved
in \(O(T^3)\) if the production capacity is constant \[8\] \[19\]. ULS with variable inventory bounds and fixed-charge holding costs can be solved in \(O(T^2)\) time \[2\] \[3\].

The rest of the paper is organized as follows. In Section \[2\], we introduce the general DP formulation for CFG, the main idea for the polynomiality proof and a key lemma based on the planarity of the grid. In Section \[3\] we show that the DP formulation can be solved in polynomial time if CFG has fixed \(L\), one echelon of sources, one echelon of sinks and \(O(1)\) different capacity values, or CFG has two echelons, \(O(1)\) different capacity values on downward arcs, and general capacities on forward arcs. We then present several NP-hard cases when some conditions are relaxed. In Section \[4\] we show that uncapacitated CFG with fixed \(L\) and one echelon of sources (or sinks) can be solved in polynomial time, and then present several NP-hard cases when any of the restrictions is removed. Conclusions are given in Section \[5\].

### 2 The DP formulation and its complexity

We formulate CFG as a discrete-time DP with \((T + 1)\) stages. Different from those in the lot-sizing literature \[8\] \[20\] \[13\], the proposed DP formulation can handle multiple echelons, general arc capacities and arbitrary distribution of sources and sinks. Another advantage is that it can deal with backward arcs easily by augmenting the state space while the one introduced in \[12\] cannot. The elements of the DP formulation are as follows:

1. Decision stages. There are \(T + 1\) stages corresponding to time period \(t = 1, \ldots, T\) with stage 0 for the dummy period 0.

2. States. Define the state \(s^t\) at stage \(t\) to be a \(L\)-dimensional vector whose component \(s^t_l\) denotes the flow over the forward arc \((v_l,t, v_l,t+1)\). We assume that each component of \(s^0\) and \(s^T\) is 0. Note that the dimension of \(s^t\) can be reduced by one since the summation of the components of \(s^t\) is always \(\sum_{l=1}^{L} b(v_l,t)\) by the flow balance constraints. When backward arcs are present, we augment the state vector \(s^t\) to include the flow over the backward arc \((v_{l,t+1}, v_{l,t})\) for each \(l\).

3. Decision variables. The decision variable \(u^t\) at stage \(t\) is a \((L - 1)\)-dimensional vector whose component \(u^t_l\) denotes the flow over the downward arc \((v_{l,t+1}, v_{l+1,t+1})\).

4. The system equations. The state \(s^{t+1}\) at stage \(t + 1\) can be easily calculated by the flow balance constraints of the nodes at time period \(t + 1\). Let the system equations be \(s^{t+1} = H_t(s^t, u^t)\), where \(H_t\) is the affine function representing the flow balance constraints for nodes at stage \(t + 1\).

5. The cost function. The cost at stage \(t\) is the sum of all costs incurred by the downward arcs and forward arcs (and backward arcs) at that stage.

Although flexible in modeling, the proposed DP formulation is difficult to solve directly since the state space at each stage is an uncountable set. From \[1\], we observe that CFG is to minimize a concave function over a flow polyhedron \(P_F\) defined by the constraints in \[1\], so the optimum must be attained at an extreme point of \(P_F\) for any
feasible instance. We call an extreme point of $P_F$ an extreme flow. Since there are only a finite number of extreme flows for a given $P_F$, it suffices to consider a finite set of states corresponding to those extreme flows in the DP formulation.

**Proposition 2.1.** CFG can be solved in $O(TK^2L)$ time, where $K$ is an upper bound for the cardinality of the set of values the flow on an arc can take for all possible extreme flows.

**Proof.** Since each component of a state vector in the DP formulation is the flow over a forward arc given some extreme flow and each state vector is $L$-dimensional, the state space at each stage of the DP has a cardinality of $O(KL)$, and there are $O(K^2L)$ actions between each consecutive stages. Then the DP can be solved by finding a shortest path over an acyclic graph with $O(TKL)$ nodes and $O(TK^2L)$ edges, which can be done in $O(TK^2L)$ time.

By Proposition 2.1, our main task is reduced to show that the value $K$ is polynomial in $T$ for CFG under certain conditions. In the literature, two techniques have been proposed to prove the polynomial solvability of a DP formulation of a lot-sizing problem. The first is the regeneration interval technique used in most lot-sizing literature: decompose the extreme flow into a set of disjoint regeneration intervals or sub-plans, argue that the subproblem in each regeneration interval or sub-plan can be solved in polynomial time, and then the original problem can be solved by finding the shortest path problem over an acyclic graph; the second is the flow decomposition technique developed in [12]: decompose the extreme flow into flows along a set of paths and argue that the flow along certain paths can only take a polynomial number of values. Our proof technique in this paper is quite different from these two, since it relies only on the facts that the grid network is planar and sources lie above sinks. To be specific, in the rest of the paper we use “path” to denote a path in the underlying undirected graph of the grid $G$ and “directed path” to denote a path in the directed graph $G$. In a directed path, the directions of the arcs must be consistent; while in a path, the directions do not matter. We first present a technical lemma relying on the planarity of the grid.

**Lemma 2.2.** Given four nodes $v^1, v^2, v^3$ and $v^4$ lying clockwise on the boundary of the grid $G$, let $P_1$ be any path connecting $v^1$ to $v^3$ and $P_2$ be any path connecting $v^2$ to $v^4$, then $P_1$ and $P_2$ must intersect.

**Proof.** Suppose that $P_1$ and $P_2$ do not intersect. Add a node $v^5$ in the outer face of $G$, and add arcs $(v^i, v^5)$ for $i = 1, 2, 3, 4$ in a way that these arcs do not intersect. Then the new graph $G' = (V \cup \{v_5\}, A \cup \cup_{i=1}^4(v^i, v^5))$ is still planar, but it contains complete graphs on five nodes (induced by nodes $v^1, v^2, \ldots, v^5$) as a minor, a contradiction to Wagner’s theorem [5].

### 3 CFG with capacities

#### 3.1 Polynomial solvable cases

Our main result is:
**Theorem 3.1.** CFG with fixed $L$, one echelon of sources, one echelon of sinks, and $O(1)$ different capacity values can be solved in polynomial time in $T$ and the number of queries of the value oracle.

**Proof.** WLOG, we can assume sources are at the first echelon and sinks are at the last echelon. Indeed, our proof only requires that all sources and sinks are at the boundary of the grid.

Given an extreme flow $f$, we say an arc $a \in A$ is tight if its flow $f_a$ equals its capacity $U_a$. Otherwise, we say the arc $a$ is untight. It is easy to see that the untight arcs in the support of $f$ form a forest (when ignoring the directions). We build a spanning tree $T = (V, A_T \subseteq A)$ of $V$ as follows. First, we add all untight arcs in the support of $f$ to $A_T$. Then, we add other arcs in $A$ to $A_T$ in arbitrary order, as long as $T$ remains a forest. At the end of the process, we obtain a spanning tree $T$ of of $V$. All untight arcs in the support of $f$ are included in $T$. Besides these arcs, some tight arcs or arcs with zero flow might be in $T$.

Given an arc $a = (u, v)$ in $T$, we are interested in the number of values it can take under all extreme flows. Removing $a$ from $T$ will break $T$ into two sub-trees: one sub-tree $T_1$ containing $u$ and the other sub-tree $T_2$ containing $v$. We say a node $v_{\ell,i} \in V$ is of type 1 if $v_{\ell,i} \in V_{T_1}$ and type 2 if $v_{\ell,i} \in V_{T_2}$. Let $b_{\ell,t}$ denote the supply $b(v_{\ell,t})$ for simplicity. By the flow balance constraints, we have $f_a = \sum_{v_{\ell,i} \in V_T} b_{\ell,t} - F$, where $F$ is the total net flow sent from $V_{T_1}$ to $V_{T_2}$ through arcs other than $a$ (if an arc sends a flow from $V_{T_2}$ to $V_{T_1}$, we take the negative value of the flow). For convenience, let $A'$ denote this set of arcs. Note that the same idea of constructing a spanning tree based on the support of the extreme flow $f$ and calculating flow $f_a$ from the net supply in one sub-tree will be used in later proofs for two-echelon CFG with $O(1)$ capacity values on downward arcs and uncapacitated CFG.

We try to bound the number of possible values for $\sum_{v_{\ell,t} \in V_{T_1}} b_{\ell,t}$ and $F$ respectively. First consider the term $\sum_{v_{\ell,t} \in V_{T_1}} b_{\ell,t}$. Since all sources and sinks are at the boundary of the grid, it suffices to consider whether each boundary node is in $V_{T_1}$ or not. The set of boundary nodes in $V_{T_1}$ must appear consecutively on the boundary. Otherwise assume that four nodes $v^1, v^2, v^3, v^4$ lie clockwise on the boundary of the grid and $v^1, v^3 \in V_{T_1}$ and $v^2, v^4 \in V_{T_2}$. Then the path connecting $v^1$ and $v^3$ in $V_{T_1}$ must intersect with the path connecting $v^2$ to $v^4$ in $V_{T_2}$ by Lemma 2.2, contradicting to the fact that the four nodes are in two disconnected sub-trees. Thus we can find two nodes $v'$ and $v''$ on the boundary such that, if we walk from $v'$ to $v''$ clockwise along the boundary, then the set of boundary nodes in $V_{T_1}$ is exactly the set of nodes in the path. Thus, $v'$ and $v''$ completely determines this set and therefore the sum. The number of possibilities for $\sum_{v_{\ell,t} \in V_{T_1}} b_{\ell,t}$ is at most $O(T^2)$.

Now we consider the term $F$. Notice that all arcs in $A'$ are either tight arcs or arcs with zero flow, since the tree $T$ contains all untight arcs. For each capacity value, we only need to know the net number of tight arcs in $A'$ from $V_{T_1}$ to $V_{T_2}$ with that capacity value. The numbers for all capacity values determine $F$. Since there $O(1)$ different capacity values, the number of possibilities for $F$ is $O(LT)^{O(1)} = T^{O(1)}$ for fixed $L$.

The restriction on arc capacities and distribution of sources and sinks in Theo-
Proposition 3.2. For $L = 2$, CFG with $O(1)$ different capacity values on downward arcs and general capacities on forward arcs can be solved in polynomial time in $T$ and the number of queries of the value oracle.

Proof. The proof idea is similar to that of Theorem 3.1. Given an extreme flow $f$, construct a spanning tree $T$ of $V$ in the same way as in Theorem 3.1. Then given a forward arc $a$ in $T$, the flow $f_a = \sum_{v \in V_2} b_{k,t} - F$, and we bound the number of possible values for $\sum_{v \in V_2} b_{k,t}$ and $F$ respectively. The term $\sum_{v \in V_2} b_{k,t}$ can take $O(T^2)$ values as in the proof of Theorem 3.1. For term $F$, we decompose it to $F_1 + F_2$, where $F_1$ is the total net flow send through downward arcs, and $F_2$ is the total net flow through forwards arcs other than $a$. The term $F_1$ can take $T^{O(1)}$ values since there are $O(1)$ different capacity values on downward arcs. The term $F_2$ can take $O(T)$ different values since there can be at most one forward arc other than $a$ sending flow from $V_1$ to $V_2$. Otherwise if there are two such forwards arcs, then among the end nodes of these two arcs and arc $a$, we can find four nodes of alternating types lying clockwise on the boundary of the grid, a contradiction to Lemma 2.2. □

3.2 NP-hard cases

In this section, we will show that the problems in Theorem 3.1 and Proposition 3.2 are likely the most general polynomial solvable cases for capacitated CFG unless P=NP.

Theorem 3.3. If there are two echelon of sinks, CFG is NP-hard with $L = 3$, a single source and a single capacity value.

Proof. Reduction from the knapsack problem: given a set of $n$ items, item $i$ has value $y_i$ and cost $c_i$ and the objective is to select a subset of items with minimum cost and the total value is at least $Y$. Choose a value $B \geq \max\{y_i | i \in [n]\}$. As shown in Figure 2, consider a CFG instance with $L = 3$ echelons, $T = n$ periods, one source $v_{1,i}$ with $b(v_{1,i}) = \sum_{i=1}^n (B - y_i) + Y$, $n + 1$ sinks $v_{2,i}$ for $i \in [n]$ and $v_{3,n}$ with $b(v_{2,i}) = -(B - y_i)$ for $i \in [n]$ and $b(v_{3,n}) = -Y$; the cost over each arc $(v_{2,i}, v_{3,n})$ is always $c_i$ for nonzero flow and 0 otherwise, the cost over the rest of arcs in Figure 2 is always 0, and the costs other arcs not present are large enough so they will never be used in an optimal solution; the capacity for each downward arc is $B$; there is no capacity on other arcs. Then any feasible solution of the knapsack instance can be converted to a feasible solution of the CFG instance with the same cost, and vice versa. Indeed given a feasible solution $S \subseteq [n]$ of the knapsack instance with cost $\sum_{i \in S} c_i$, construct a feasible flow in the following way: the flow over the arc $(v_{1,i}, v_{2,i})$ is exactly the demand $(B - y_i)$ at $v_{2,i}$ for $i \notin S$. Use arcs $(v_{2,i}, v_{3,n})$ for $i \in S$ to send flow and satisfy the demand at $v_{3,n}$. The maximum amount of flow that can be sent along the arc $(v_{2,i}, v_{3,n})$ is $B - (B - y_i) = y_i$ for $i \in S$, so the maximum amount of flow that can be sent through arcs $(v_{2,i}, v_{3,n})$ for $i \in S$ to $v_{3,n}$ is $\sum_{i \in S} y_i \geq Y$. The flow has costs $\sum_{i \in S} c_i$. Conversely, given a feasible flow in the CFG instance, set $S = \{i | \text{the flow over arc } (v_{2,i}, v_{3,n}) \text{ is nonzero}\}$, then $S$ is a feasible solution of the knapsack instance with the same cost. □
Figure 2: Three-echelon CFG instance with two echelons of sinks, a single source and a single capacity value. (The amount of supply or demand is marked beside each node. The pair \((c, U)\) on some arc \(a\) denotes that the cost is \(c\) if the arc carries nonzero flow and 0 otherwise and the arc capacity is \(U\). An arc without such pair over it means that there is no capacity and no cost of sending flow over that arc. An arc not present in the figure means that the cost is large enough of sending any flow over that arc so it will never be used in any optimal solution. The rest of the figures in this paper will adopt the same notation.)

**Corollary 3.4.** CFG with \(L \geq 3\), a single source, a single sink, and general capacities on each forward arc is NP-hard.

**Proof.** A reduction from the knapsack problem. Given a knapsack instance we construct the three-echelon CFG instance in Figure 3 with a single source \(v_{1,1}\) and a single sink \(v_{3,2n}\). Then any feasible solution of the knapsack problem can be converted to a feasible flow of the CFG instance with the same cost, and vice versa. 

Figure 3: Three-echelon CFG instance with general capacities on forward arcs.

By Theorem 3.3 and Corollary 3.4, the multi-echelon serial supply chain problem with intermediate demands, and the multi-echelon serial supply chain problem
with variable inventory bounds are unlikely to have efficient algorithms unless P=NP. This is quite different from the uncapacitated multi-echelon lot-sizing problem with intermediate demands \cite{12} and the single-echelon lot-sizing problem with inventory bounds \cite{3}, both of which can be solved in polynomial time.

\section{Uncapacitated CFG}

\subsection{Polynomial solvable cases}

Our main result in this section is:

\begin{theorem}
If all sources lie at one echelon and the number of echelons \( L \) is fixed, CFG can be solved in time polynomial in \( T \) and the number of queries of the value oracle.
\end{theorem}

\begin{proof}
\text{Theorem 4.1} follows directly from Proposition 4.2 below and backward induction for the DP formulation in Section 3.
\end{proof}

\begin{proposition}
If all sources lie at one echelon and the number of echelons \( L \) is fixed, the set of values of the flow on each arc for all possible extreme flows is polynomial in \( T \).
\end{proposition}

Proposition 4.2 presents a stronger result than what we need. It shows that each arc (not only the forward arcs in the state vector) can only take a polynomial number of values under all extreme flows of CFG.

We now prove Proposition 4.2. Given any extreme flow \( f \) of an uncapacitated network, it is known that the support of \( f \) forms a forest. We now construct a spanning tree \( T \) as in the capacitated case as follows. We add all nodes \( V \) and all arcs \( a \) in the support of \( f \) to \( T \). We say a node \( v \in V \) can be reached from the first echelon in \( T \), if there is a path in \( T \) connecting a first-echelon node to \( v \) such that the directions of the arcs in the path are towards \( v \). As long as the underlying undirected graph of \( T \) is not a spanning tree, we add to \( T \) some arc \( a = (u, v) \in A \setminus T \) satisfying the following two conditions: (1) the underlying undirected graph of \( T \cup \{a\} \) is a forest; (2) \( u \) can be reached from the first echelon in \( T \). Notice that initially, all non-isolated nodes (that is, nodes incident to at least one arc in \( T \)) can be reached from the first echelon. Thus, by maintaining this invariant, we can make sure that the final graph \( T \) is a spanning tree of \( V \) (ignoring the directions of arcs). Up to now, we have constructed a spanning tree \( T = (V, A') \) satisfying the following two conditions:

\begin{enumerate}
\item[(A1)] All arcs in the support of \( f \) are in \( A' \).
\item[(A2)] For every node \( v \in V \), there is a directed path in \( T \) connecting some first-echelon node to \( v \).
\end{enumerate}

For the remainder of the proof we will focus on some arc \( a = (u, v) \in A' \) and show that the set of values that \( f_a \) can take is polynomial in \( T \) under all extreme flow \( f' \)'s. Removing \( a \) from \( T \) will result in two sub-trees of \( T \): the sub-tree \( T_1 \) containing \( u \) and the sub-tree \( T_2 \) containing \( v \). Then, we have \( f_a = -\sum_{(v, e) \in V_2} b_{e,i} \). It suffices
to consider the set $V_{T_2}$ of nodes in $T_2$. The following lemma immediately implies Proposition 4.2.

**Lemma 4.3.** For each $\ell \in [L]$, the set $\{i \in [T] : v_{\ell,i} \in V_{T_2}\}$ is the union of at most three intervals. In other words, there are six integers $i_1, j_1, i_2, j_2, i_3, j_3$ such that $\{i \in [T] : v_{\ell,i} \in V_{T_2}\} = \{i : i_1 \leq i \leq j_1 \text{ or } i_2 \leq i \leq j_2 \text{ or } i_3 \leq i \leq j_3\}.$

**Proof.** Recall that a node $v_{\ell,i} \in V$ is of type 1 if $v_{\ell,i} \in V_{T_1}$ and type 2 if $v_{\ell,i} \in V_{T_2}$. Thus, the $T$ nodes at each echelon $\ell$ is partitioned into many intervals, each being a maximal interval of nodes with the same type. Our goal is to show that there are at most 3 type 2 intervals for each echelon $\ell$. We use an example to illustrate the idea of proof. Given a spanning tree $T$ and an arc $(u,v)$ in Figure 4, nodes at echelon $\ell$ are divided into two types: $v_{1,i_2}, v_{1,i_3}$ and $v_{1,i_6}$ are of type 1 and nodes $v_{1,i_1}, v_{1,i_4}, v_{1,i_5}$ and $v_{1,i_7}$ are of type 2.

![Figure 4](image.png)

Figure 4: Given a spanning tree $T$ and an arc $(u,v)$, the nodes at echelon $\ell$ are divided into type 1 (hollow dots) and type 2 (solid dots).

First focus on the case $\ell = 1$.

**Claim 4.4.** There cannot be four nodes $v_{1,i}, v_{1,j}, v_{1,i'}, v_{1,j'}$, $1 \leq i < j < i' < j' \leq T$ with alternating types in that order.

**Proof.** Follows from Lemma 2.2.

Thus, we have proved the lemma for the case $\ell = 1$. Now, consider the harder case $\ell > 1$. To prove the lemma for this case, we shall use the crucial fact that all sources are at the first echelon. By Property (A2), there is a directed path in $T$ connecting some first-echelon node to $v_{\ell,i}$ for every $i \in [T]$. For each $i \in [T]$, define $\kappa_i$ to be the smallest integer $j \in [T]$ such that there is a directed path connecting $v_{1,j}$ to $v_{\ell,i}$ in $T$. Define $P_i$ to be the unique directed path in $T$ connecting $v_{1,\kappa_i}$ to $v_{\ell,i}$. In Figure 4 $\kappa_{i_1} = i_1, \kappa_{i_2} = \kappa_{i_3} = \ldots = \kappa_{i_6} = i_2$ and $\kappa_{i_7} = i_3$.  

11
Claim 4.5. \( \kappa : [T] \to [T] \) is a non-decreasing function. Moreover, the union of paths \( \{ P_i : i \in [T] \} \) form a forest of arborescences.

Proof. Assume for some \( i < i' \) we have \( \kappa_i = j > \kappa_{i'} = j' \). Then consider the two directed paths \( P_i \) and \( P_{i'} \) in \( T \). Since there are only forward and downward arcs, the two paths must intersect. Then it follows that there is a directed path connecting \( v_{1,j'} \) to \( v_{1,i} \) in \( T \), contradicting the definition of \( \kappa_i \).

For the second statement, it suffices to prove that if \( P_i \) and \( P_{i'} \) intersect then they start from the same first-echelon node. Assume they start from two different nodes \( v_{1,i} \) and \( v_{1,j'} \) and \( i < j' \). Then since they intersect, there is a directed path connecting \( v_{1,i'} \) to \( v_{1,j} \), contradicting the definition of \( P_j \).

We further categorize type 2 nodes in echelon \( \ell \) into two types. We say a node \( v_{\ell,i} \) is of type 2A if \( P_i \) contains the arc \( a = (u,v) \); otherwise, we say \( v_{\ell,i} \) is of type 2B. Notice that if \( v_{\ell,i} \) is a type 1 node, then \( P_i \) does not contain \( a = (u,v) \) since \( u \) is of type 1 and \( v \) is of type 2. In Figure 4, \( v_{\ell,i_4} \) and \( v_{\ell,i_6} \) are of type 2A, and \( v_{\ell,i_1} \) and \( v_{\ell,i_7} \) are of type 2B.

Claim 4.6. If \( v_{\ell,i} \) and \( v_{\ell,j} \) are of type 2A and \( 1 \leq i < k < j \leq T \), then \( v_{\ell,k} \) is of type 2A.

Proof. Since both \( P_i \) and \( P_j \) contains \( a \), by the second statement of Claim 4.5, we have \( \kappa_i = \kappa_j \). Then by the first statement of Claim 4.5, we have \( \kappa_i = \kappa_k = \kappa_j \). From the topology structure of \( P_i \cup P_k \cup P_j \), \( P_k \) must contain \( a \).

Thus, the set of type 2A nodes at echelon \( \ell \) form an interval.

Claim 4.7. There cannot be four integers \( 1 \leq i < j < i' < j' \leq T \) with \( v_{\ell,i} \) and \( v_{\ell,i'} \) being of type 1 and \( v_{\ell,j} \) and \( v_{\ell,j'} \) being of type 2B.

Proof. Consider the four directed paths \( P_i, P_j, P_{i'}, P_{j'} \). None of the four directed paths contains \( a \). Then \( v_{1,\kappa_i}, v_{1,\kappa_j}, v_{1,\kappa_{i'}}, v_{1,\kappa_{j'}} \) are of type 1, type 2, type 1, type 2 respectively. Moreover, \( \kappa_i < \kappa_j < \kappa_{i'} < \kappa_{j'} \) by Claim 4.5. This contradicts Claim 4.4.

Similarly, it can not happen that \( v_{\ell,i} \) and \( v_{\ell,i'} \) are of type 2B and \( v_{\ell,j} \) and \( v_{\ell,j'} \) are of type 1.

With all these claims, we are ready to finish the proof of Lemma 4.3. If echelon \( \ell \) does not contain type 2A nodes, then there can be at most two maximal intervals of type 2B nodes, by Claim 4.7. If echelon \( \ell \) contains type 2A nodes, then by Lemma 4.6, these nodes form an interval. Removing this type 2A interval from \( [T] \) will break \([T]\) into two intervals. In Figure 4, the two intervals are \([i_3,i_5]\) and \([i_6,i_7]\). By Claim 4.7, each of the two intervals contains at most two maximal type 2B intervals. Moreover, if some interval contains two maximal type 2B intervals, then one of the type 2B intervals must be adjacent to the type 2A interval and we can merge them into one type 2 interval. Overall, there are at most 3 maximal type 2 intervals, finishing the proof of Lemma 4.3.
4.2 NP-hard cases

Each restriction in Theorem 4.1 turns out to be crucial for the polynomial solvability of the problem in Theorem 4.1, as shown by a series of NP-hard cases below.

**Proposition 4.8.** CFG with two echelons of sources and two echelons of sinks is NP-hard.

**Proof.** Reduction from the partition problem. An instance of the partition problem asks that given a set $S$ of integers $y_1, \ldots, y_n$ whether there exists a partition of $S$ such that the sum of the numbers in each partition is equal to $\sum_{i=1}^{2n} y_i / 2$. We construct a CFG instance with two echelons of sources on top of two echelons of sinks. Let $D$ be a number larger than $\sum_{i=1}^{n} y_i$. Consider the grid network in Figure 5 with $L = 4$ echelons, $T = 2n$ periods, $n + 1$ sources $v_{1,1}$ and $v_{2,2n-1}$ ($i \in [n]$), and $n + 1$ sinks $v_{3,2i}$ ($i \in [n]$) and $v_{4,2n}$. Then the minimum cost of that instance is $n$ if and only if the partition instance is a yes instance and $n + 1$ otherwise.

![Figure 5: CFG instances with two echelons of sources on top of two echelons of sinks.](image)

**Proposition 4.9.**

1. If the number of echelons $L$ is an input parameter, CFG is NP-hard.

2. If the grid network contains upward arcs, CFG is NP-hard for fixed $L \geq 3$.

**Proof.** Part 1. A reduction from the partition problem. Consider the grid network in Figure 6 with $L = n + 1$ echelons, $T = n + 1$ periods, two sources $v_{1,1}$, $v_{1,2}$ and $n$ sinks $v_{i+1,j+1}$’s for $i \in [n]$. Then the minimum cost of that CFG instance is $n$ if and only if the partition instance is a yes instance and $n + 1$ otherwise.

Part 2. A reduction from the partition problem. Consider the grid network in Figure 7 with $L$ echelons ($L \geq 3$), $T = n + 1$ periods, two sources $v_{1,1}$, $v_{1,2}$ and $n$ sinks for $i \in [n]$. Then the minimum cost of that CFG instance is $n$ if and only if the partition instance is a yes instance and $n + 1$ otherwise.

\[\sum_{i=1}^{n} y_i / 2\]
Corollary 4.10. CFG with $L = 3$ and an arbitrary distribution of sources and sinks is NP-hard.

Proof. Follows from a similar reduction from the partition problem as in the proof of Proposition 4.8. Consider the grid network in Figure 8 with $L = 3$ echelons, $T = 2n$ periods, $n + 1$ sources $v_{1,1}$ and $v_{2,2i-1}$ ($i \in [n]$), and $n + 1$ sinks $v_{2,2i}$ ($i \in [n]$) and $v_{3,2n}$. Then the minimum cost of that CFG instance is $n$ if and only if the partition instance is a yes instance and $n + 1$ otherwise.

5 Conclusions

In this paper, we gave a characterization of the computational complexity of the minimum concave cost flow in grid networks based on arc capacities, number of echelons...
and distribution of sources and sinks. We developed a new technique to analyze the structure of optimal solutions based on the planarity of the grid. Our results resolve several open questions raised in the literature. It should be noted that the polynomial-time algorithms developed in this paper can be easily extended to all the cases with backward arcs, by including the corresponding backward arcs in the state space of the DP formulation. One problem left open in this paper is the complexity of CFG with one echelon of sources, one echelon of sinks, $O(1)$ capacity values and general $L$.

References


