A Generalized Dilworth's Theorem, with Application to Routing and Scheduling

John N. Hooker
Carnegie Mellon University, john@hooker.tepper.cmu.edu

N R. Natraj
Carnegie Mellon University

Follow this and additional works at: http://repository.cmu.edu/tepper
Part of the Economic Policy Commons, and the Industrial Organization Commons
A Generalized Dilworth’s Theorem, with Application to Routing and Scheduling *

J. N. Hooker
N. R. Natraj
Graduate School of Industrial Administration
Carnegie Mellon University, Pittsburgh, PA 15213 USA

April 1991
Revised January 1992

Abstract

Dilworth’s theorem states a duality relation between minimum chain decompositions of a directed, acyclic graph and maximum antichains. We generalize the theorem to apply when the chains of the decomposition are required to contain the chains of an initial decomposition. We show that duality obtains precisely when an associated undirected graph is perfect. We apply this result to a vehicle routing and scheduling problem with time windows. Here each chain of the initial decomposition contains nodes that correspond to the pickup, delivery and possibly intermediate stops associated with a piece of cargo.

1 Introduction

A chain decomposition of a directed, acyclic graph is a partition of its nodes into chains (directed paths). An antichain is a set of nodes one in which no pair of nodes is connected by an arc. A

classic theorem of R. P. Dilworth [3] states that the minimum number of chains into which the graph can be decomposed is equal to the size of a maximum antichain.

This theorem has been generalized primarily along two directions. Taking one direction, Ford and Fulkerson showed [5] that it is a special case of the max cut/min flow theorem for networks. Taking another direction, C. Greene and D. L. Kleitman [6] generalized the idea of an antichain to cover node sets containing chains of length at most \( k \); classical antichains have \( k = 1 \).

We generalize Dilworth's theorem in a third direction by requiring that a chain decomposition improve on a given decomposition. That is, we begin with an initial decomposition containing many chains and try to combine them into fewer chains. We call this the chain merger problem. It reduces to the classical chain decomposition problem when the initial chains consist of single nodes.

We show that Dilworth's duality relation holds for the chain merger problem precisely when an associated undirected graph is perfect. In fact the chain merger problem can be solved by solving a minimum clique covering problem on the associated graph, or a coloring problem on its complement. This problem is soluble in polynomial time when the graph is perfect, which is to say when the duality relation holds.

The chain merger problem was suggested to us by a vehicle routing and scheduling problem with time windows. Each piece of cargo to be delivered is associated with nodes representing its pickup, delivery, and possibly intermediate stops for processing or inspection. An arc connects one node with another when it is possible for a vehicle to move from one point to the other within the associated time windows. If the problem has a property we call "chain feasibility," it is a chain merger problem. Each initial chain contains the pickup, delivery and intermediate stops associated with one piece of cargo. After the problem is solved, each chain contains the stops to be made by a single vehicle. The object is to minimize the number of vehicles required and therefore the number of chains.

In the first section below we prove the generalized Dilworth’s theorem. In the second section we characterize chain feasible routing and scheduling problems and show how to formulate them as chain merger problems.
2 Generalized Dilworth’s Theorem

In this section we first introduce some notation and definitions. We then present a generalization of the chain decomposition problem. We end this section with a generalization of Dilworth’s theorem.

Let us consider a transitive, acyclic directed graph $G = (V,E)$, where $V$ denotes the set of nodes and $E$ denotes the set of arcs. A chain in $G$ is denoted by a sequence of distinct nodes $(j_1, \ldots, j_n)$ such that $(j_k, j_{k+1}) \in E$ for all $k = 1, \ldots, n-1$. We are given an initial partition of $V$ into chains $C_1, \ldots, C_m$. The classical chain decomposition problem [1, 2] corresponds to the case when $|C_1| = \ldots = |C_m| = 1$ and $m = |V|$.

A chain merger is a partition of the nodes of $G$ into chains such that each chain is admissible; i.e., it consists of the nodes in one or more of the chains $C_1, \ldots, C_m$. The chain merger problem is the problem of finding a chain merger containing a minimum number of chains.

We say that a collection of chains is chainable if some admissible chain contains all the nodes in the corresponding chains. Given the graph $G$ and the chains $C_1, \ldots, C_m$ we construct a related undirected graph $\tilde{G} = (\tilde{V},\tilde{E})$ as follows. The nodes in $\tilde{V}$ correspond to the chains $C_i$, and two nodes $i, j \in \tilde{V}$ are connected by an edge if and only if $\{C_i, C_j\}$ is chainable.

We are now ready to state a modification of Dilworth’s theorem that generalizes a familiar duality property of the classical chain decomposition problem. The property appears in chain merger problems for which the associated graph $\tilde{G}$ is perfect. A perfect graph is one in which the maximum number of pairwise nonadjacent nodes in any node induced subgraph is equal to the minimum number of cliques required to cover the nodes of the subgraph.

For any minimum chain merger problem on graph $G$, and $S \subseteq \{C_1, \ldots, C_m\}$, let $G_S$ be the subgraph of $G$ induced by the nodes contained in the chains in $S$. We say that the chain merger problem induced by $S$ is the chain merger problem on $G_S$.

**Theorem: 1 (Generalized Dilworth’s Theorem)** For a given chain merger problem with initial chains $C_1, \ldots, C_m$, let $\alpha_S$ be the maximum number of node sets that are pairwise unchainable as admissible chains in the problem induced by $S \subseteq \{C_1, \ldots, C_m\}$. Let $\theta_S$ be the minimum number of admissible chains required to cover the nodes in the chains in $S$. Then $\alpha_S = \theta_S$ for any $S$ if and
only if the related undirected graph \( \tilde{G}_S \) is perfect.

In order to prove the theorem we need to show that the chain merger problem on \( G \) can be solved as an appropriate problem on the related undirected graph \( \tilde{G} \). In fact, the following theorem and its corollary prove that it can be solved as a minimum clique covering problem on \( \tilde{G} \).

**Lemma: 1**  Given an acyclic, transitive graph \( G = (V, E) \) and chains \( C_1, \ldots, C_m \), the set \( \{C_1, \ldots, C_m\} \) is chainable if and only if every pair \( \{C_i, C_j\} \) is chainable.

**Proof:** Assume first that \( \{C_1, \ldots, C_m\} \) is chainable. The chainability of every pair follows from the transitivity of the graph \( G \).

Now suppose every pair \( \{C_i, C_j\} \) of \( \{C_1, \ldots, C_m\} \) is chainable. The chain \( C_{ij} \) associated with \( \{C_i, C_j\} \) imposes an order on the nodes in \( C_i \cup C_j \). Now we construct the admissible chain \( C \) that chains together all the nodes in \( \{C_1 \cup \ldots \cup C_m\} \) by applying the following procedure:

0. Start with an empty chain \( C \) and let \( \bar{C} = V \).

1. Form the list of nodes that occur first in the chains \( C_{ij} \).

2. From this list find the unique node \( c \) that does not have any other node in the list as its predecessor in any of the chains. Add the node \( c \) to the end of \( C \) and remove \( c \) from \( \bar{C} \). Also remove \( c \) and the arc that leaves \( c \) in all chains \( C_{ij} \) containing \( c \).

3. Repeat steps 1 and 2 until \( \bar{C} \) is empty.

Since \( G \) is acyclic, node \( c \) always exists. It is also unique because if two nodes \( i, j \) had no predecessors, then neither \( (i, j) \) nor \( (j, i) \) would exist, which means (by transitivity) that the node sets containing \( i \) and \( j \) would not be chainable, a contradiction. Clearly the procedure is finite and terminates with the desired chain \( C \).

Given an undirected graph \( \tilde{G} = (\tilde{V}, \tilde{E}) \), the clique covering problem is to find a collection of subsets \( \tilde{V}_1, \ldots, \tilde{V}_k \) of \( \tilde{V} \) such that \( \bigcup_{i=1}^{k} \tilde{V}_i = \tilde{V} \), each \( \tilde{V}_i \), \( (i = 1, \ldots, k) \) induces a complete subgraph of \( \tilde{G} \), and the number \( k \) of such subsets is as small as possible. If in addition \( \tilde{V}_i \cap \tilde{V}_j = \emptyset \) for all \( i \neq j \) we have the clique partitioning problem. Given a clique cover of \( \tilde{G} \) one can obtain a clique
partition of the same size simply by removing nodes that belong to more than one subset \( V_i \), from all but one of the subsets. Hence solving a clique covering problem solves the clique partitioning problem also.

**Corollary: 1**  The chain merger problem on a transitive, acyclic graph can be solved as a minimum clique covering problem on the graph \( \bar{G} \).

**Proof:** Lemma 1 implies that there is a one-to-one correspondence between admissible chains in the graph \( G \) and cliques in \( \bar{G} \). Hence a chain merger of \( G \) corresponds to covering the nodes of \( \bar{G} \) with a minimum number of cliques.

The node coloring problem is to color the nodes of a graph with a minimum number of colors so that no two adjacent nodes have the same color. The number of colors required is the chromatic number of the graph. It is obvious that solving the minimum clique covering problem on \( \bar{G} \) is equivalent to finding the chromatic number of the complement of \( \bar{G} \). Hence the chain merger problem can also be solved as a node coloring problem on the appropriate graph [4, 12].

**Proof of Theorem 1:** The theorem follows immediately from Corollary 1 and the definition of perfect graphs.

It should be noted that when \( |C_1| = \ldots = |C_m| = 1 \) and \( m = |V| \), Theorem 1 is Dilworth’s theorem [3]. In this case the related undirected graph \( \bar{G} \) is the comparability graph and it is known to be perfect [7]. Furthermore, the chain merger problem is the classical chain decomposition problem, which can be solved in polynomial time using bipartite matching [5].

More generally, any chain merger problem for which \( \bar{G} \) is perfect can be solved in polynomial time. This is because the equivalent clique covering problem on \( \bar{G} \) can be solved as a linear programming problem [8].

### 3 Application to a Routing and Scheduling Problem

We now show how certain routing and scheduling problems may be formulated as chain merger problems.
Suppose that \( m \) cargo items must be transported, and each item \( i \) must make stops \( v_{i1}, \ldots, v_{ip_i} \). Typically \( v_{i1} \) is the pickup, \( v_{ip_i} \) is the delivery, and the rest are intermediate stops. The object is to route and schedule as few vehicles as possible to make the stops within time windows given for each stop.

The stops \( v_{iq} \) for all cargo items \( i \) can be regarded as the nodes \( v_1, \ldots, v_n \) of a precedence graph \( G' \). Associated with each node \( v_j \) is a time window \([a_j, b_j]\), which indicates that the cargo must arrive at \( v_j \) no later than \( b_j \) and can leave no earlier than \( a_j \). \( G' \) contains an arc \((v_j, v_k)\) when it is possible for a vehicle to leave \( v_j \) at time \( a_j \) and arrive at \( v_k \) by time \( b_k \). \( G' \) need be neither transitive nor acyclic, but we will address this issue shortly.

We have an initial decomposition of the nodes into chains \( C_1, \ldots, C_m \), where each \( C_i \) contains the stops \( v_{i1}, \ldots, v_{ip_i} \) for cargo item \( i \). If a separate vehicle delivered each cargo item, \( m \) vehicles would be required. We may be able to combine these initial chains into fewer chains, each containing the stops to be made by a single vehicle. Thus by solving the chain merger problem, we minimize the number of vehicles required.

Unfortunately, a vehicle may not be able to make the series of stops belonging to a single chain without violating the time windows. This is true even though it can move from any given stop to the next within the time windows (otherwise the connecting arc would not be present). To see this, consider three stops \( v_1, v_2, v_3 \) that are arranged geographically in a straight line. Either of the two connecting links requires one hour of travel time. The time windows are \([a_1, b_1] = [a_2, b_2] = [7 \text{ am}, 9 \text{ am}]\) and \([a_3, b_3] = [6 \text{ am}, 8 \text{ am}]\). The vehicle can move from \( v_1 \) to \( v_2 \) within the time windows by leaving \( v_1 \) no later than 8 am, and it can move from \( v_2 \) to \( v_3 \) by leaving \( v_2 \) at 7 pm. But if it begins at \( v_1 \), it will not arrive at \( v_2 \) until 8 am and therefore cannot reach \( v_3 \) by 8 am.

This example also shows that \( G' \) need be neither transitive nor acyclic. It is nontransitive because \((v_1, v_3)\) is not an arc. It is not acyclic because both \((v_1, v_2)\) and \((v_2, v_1)\) are arcs.

There is a class of routing and scheduling problems, however, in which a vehicle can always make the stops in any chain without violating the time windows. We call these chain feasible problems, which we will characterize more fully at the end of this section. First we focus on defining chain feasibility and showing that chain feasible problems can be solved as chain merger problems.
Let $t_{jk}$ be the travel time from $v_j$ to $v_k$. A chain $(v_1, \ldots, v_q)$ is time feasible if a single vehicle can make the stops in it; that is, if $T_{br} \leq b_r - a_1$ for $r = 2, \ldots, q$, where
\[ T_{rs} = \sum_{u=r}^{s-1} t_{u,u+1}. \]

A routing and scheduling problem is chain feasible if every chain of $G'$ (admissible or otherwise) is time feasible.

**Theorem:** A chain feasible routing and scheduling problem with precedence graph $G'$ can be solved as a chain merger problem on $G'$, in the following sense: the chains in any solution of the chain merger problem indicate the routings of the vehicles in some optimal solution of the routing and scheduling problem.

**Proof:** We first note that since the routing and scheduling problem is chain feasible, $G'$ is transitive. We will construct a transitive and acyclic graph $G$ with the property that any solution of the chain merger problem on $G$ corresponds to a routing that minimizes the number of vehicles.

The nodes of $G'$ are partitioned into dicomponents, which are maximal node sets in which every pair of nodes is connected by a chain in both directions. Since $G'$ is transitive, every pair of nodes in a dicomponent is linked by an arc in both directions. Number the nodes of each dicomponent in arbitrary order, but making sure that the nodes in any $C_i$ are numbered in a way that is consistent with the order they occur in $C_i$. For each pair $v_j, v_k$ of nodes in each dicomponent, remove the arc from the higher to the lower numbered node, and let $G$ be the resulting acyclic graph.

By chain feasibility, the feasible routings correspond exactly to the decompositions of $G'$ into admissible chains. Since $G$ is a subgraph of $G'$, it remains only to show that for every decomposition of $G'$ into admissible chains, there is such a decomposition of $G$ having the same number of chains.

Let $C$ be any chain in a decomposition $D$ of $G'$ into admissible chains. The intersections of $C$ with the dicomponents of $G'$ partition $C$ into subchains. If we rearrange the nodes of each subchain in ascending order (by the numbering defined above), we obtain an admissible chain of $G'$ that, by construction, is also a chain of $G$. $G$ therefore has a decomposition into the same number of admissible chains as $D$. 


From this theorem it follows that a chain feasible routing and scheduling problem can be solved in polynomial time when the associated graph \( \tilde{G} \) is perfect.

We now characterize chain feasible routing and scheduling problems. We note first that a routing and scheduling problem is chain feasible if and only if every chain \( C = (v_1, \ldots, v_q) \) of \( G' \) satisfies \( T_{1q} \leq b_q - a_1 \). Since \( T_{1q} \) is the length of \( C \), we can check a problem for chain feasibility by computing a longest chain between every pair of nodes \( v_j, v_k \) that are connected by a chain. The problem is chain feasible if and only if no such chain is longer than \( b_k - a_j \). See [11] for a discussion of longest path algorithms.

Problems on acyclic graphs \( G' \) have the interesting property that they are chain feasible if and only if every time window \( [a_j, b_j] \) can be reduced to a point \( u_j \) without changing \( G' \).

**Lemma: 2** A routing and scheduling problem on an acyclic graph \( G' \) is chain feasible if and only if there exists for each node \( v_j \) of \( G' \) a number \( u_j \in [a_j, b_j] \) such that \( t_{jk} \leq u_k - u_j \) for all arcs \( (v_j, v_k) \) of \( G' \).

**Proof:** First suppose that the numbers \( u_j \) exist. Then for any chain we can obtain a feasible schedule by letting the vehicle stop at time \( u_j \) at each node \( v_j \).

Suppose conversely that the problem is chain feasible. For any given node \( j \) and chain \( C \) let \( u_j^C \) be the latest time in \( [a_j, b_j] \) that belongs to some schedule for \( C \). Also let \( u_j = \min_C \{ u_j^C \} \), where \( C \) ranges over all chains containing \( v_j \). Clearly \( a_j \leq u_j \leq b_j \). We also claim that \( u_j \leq u_k - t_{jk} \) for all arcs \( (v_j, v_k) \). For suppose this does not hold for some arc \( (v_j, v_k) \). Since \( (v_j, v_k) \) form a chain, \( u_j \) belongs to some feasible schedule for \( (v_j, v_k) \), where the earliest arrival at node \( v_k \) for such a schedule is \( u_k' = u_j + t_{jk} > u_k \). By definition of \( u_k \), there must be a chain \( C \) containing \( v_k \) for which \( u_k \) is the latest departure time from \( v_k \). Let \( C' \) be the portion of \( C \) that includes and follows \( v_k \). Since \( G' \) is acyclic, \( C' \) does not contain \( v_j \), so that \( \{v_j\} \cup C' \) is a chain. But \( u_k' > u_k \), which means that \( u_j \) belongs to no feasible schedule for the chain \( \{v_j\} \cup C' \), contrary to the definition of \( u_j \). The lemma follows.

The lemma does not hold for graphs \( G' \) containing a cycle. For instance, consider a graph consisting only of arcs \( (v_1, v_2) \) and \( (v_2, v_1) \), with \( a_j = 0 \) and \( b_j = t_{12} = t_{21} > 0 \) for \( j = 1, 2 \).
Then both chains \( (v_1, v_2) \) and \( (v_2, v_1) \) are time feasible, but it is impossible that \( u_1 \leq u_1 - t_{12} \) and \( u_2 \leq u_1 - t_{21} \).

Lemma 2 identifies an important class of practical chain feasible problems: those in which a vehicle departs every stop \( v_j \) at a predetermined time \( u_j \) (it may arrive at \( v_j \) before \( u_j \)). In fact every chain feasible problem on an acyclic \( G' \) is equivalent to such a problem, and the proof of Lemma 2 provides an algorithm for computing the departure times \( u_j \).

One can adjust the time windows in a problem that is not chain feasible so as to make it chain feasible. One approach is to start executing a longest chain algorithm and to stop whenever an infeasible chain is discovered. Break the chain by adjusting the windows of two adjacent nodes in the chain so as to remove the arc between them. An arc \( (v_j, v_k) \) can be removed by increasing \( a_j \), decreasing \( b_j \), or both. (One would generally adjust the two windows that need the least adjustment to break the chain.) Then restart the longest chain algorithm with the adjusted data, and continue in this fashion until no infeasible chains occur.

References


