Abstract

For variational inequality (VI) models with ‘difficult’ and ‘easy’ variables in a certain sense, we derive generalized simplicial decomposition of VIs (GSD-VI), based on applying a Dantzig-Wolfe decomposition method for VIs. In the most general case, the VI is decomposed into a master VI and a VI subproblem. Applied to a
semi-asymmetric VI problem, the subproblem is a nonlinear program (NLP); the convergence proof requires no monotonicity assumption about the asymmetric part of the VI mapping for the ‘difficult’ variables, but only requires that the subproblem’s objective function is convex. Another special case of GSD-VI reduces to a typical simplicial decomposition method, i.e., the corresponding subproblem is a linear program (LP); convergence requires no assumptions about monotonicity of the VI mapping.

Several computational tests are performed, using various semi-asymmetric traffic assignment problems (TAPs). Based on the tests, GSD-VI is faster than the reference method (relaxation method) when the number of asymmetric functions in the model is small. Comparing GSD-VI (with an NLP subproblem) to simplicial decomposition (with an LP subproblem), GSD-VI takes far fewer decomposition steps. In addition, we study the effectiveness of the column dropping technique and of approximate solution of the master problem (increasing in accuracy with decomposition steps) for the simplicial decomposition special case; the approximate master does not improve computational efficiency, but column dropping sometimes does although unreliably.

Simplicial decomposition – Variational inequalities – Dantzig-Wolfe decomposition
1 Introduction

Von Hohenbalken (1977) presented a simplicial decomposition (SD) method for nonlinear programming problems and mentioned that the SD method belongs to a class of decomposition methods in another sense, namely those descending from the decomposition principle for linear programs (LPs) of Dantzig and Wolfe (1960). Patriksson (1999, chapter 9) brought the SD method closer to Dantzig-Wolfe decomposition (DW) method by dualizing difficult constraints in the LP subproblem with prices provided by the master problem. These works of von Hohenbalken and Patriksson showed the close relationship between the DW and SD methods in LP and NLP problems, but not in variational inequalities (VIs).

Lawphongpanich and Hearn (1984) discussed a SD method for VIs in the context of how to solve asymmetric Traffic Assignment Problems (TAPs), here denoted LH-SD of VI. They provided the convergence properties of the resulting algorithm, which included approximation of the master problem that is forced to get more precise with the iteration count, and a column dropping scheme to control the size of the master problem. García et al. (2003) presented a column generation algorithm for VIs in which the subproblem is defined as an NLP.

Lawphongpanich and Hearn (1990) presented a Benders decomposition method for VIs and showed an application to the TAP. They considered that the decomposition method was especially effective for problems with which were partially asymmetric (termed semi-asymmetric, here). Examples of such semi-asymmetric problems can be found in trans-
portation network equilibrium models called combined models (see Boyce (2007a, 2007b) and de Cea et al. (2005)). In their TAP example, Lawphongpanich and Hearn (1990) followed the usual approach of Benders decomposition, by putting the ‘difficult’ variables (with asymmetric cost functions) in the master problem, and the ‘easy’ variables (symmetric costs) in the subproblem.

On the other hand, Fuller and Chung (2005) and Chung et al. (2006) showed how to apply DW decomposition to VIs. In this paper, we describe a general method of applying DW decomposition to VIs which have ‘difficult’ variables and ‘easy’ variables in a certain sense, to derive what we call Generalized Simplicial Decomposition for VIs (GSD-VI). Applied to a semi-asymmetric VI (e.g., a TAP), GSD-VI has a subproblem that is an NLP, except in the special case that all variables are difficult, which produces an LP subproblem that is identical to the one in LH-SD of VI.

Except for the master approximation and column dropping, GSD-VI with an LP subproblem is the same algorithm as LH-SD of VI, but we prove convergence with no monotonicity conditions on the VI mapping, whereas Lawphongpanich and Hearn (1984) assume strong monotonicity. In Section 4 of this paper, we study the effectiveness of master approximation and column dropping in LH-SD of VI for several TAPs.

The form of the NLP subproblem in GSD-VI differs from the forms discussed by García et al. (2003), and again, we prove convergence under more general conditions on the VI mapping. When applied to the semi-asymmetric TAP, with the easy variables corresponding to the symmetric part of the VI mapping, we only need to assume that the symmetric part is the gradient of a convex function, to ensure convergence; the asymmetric
part need only be continuous.

Unlike the Benders decomposition of Lawphongpanich and Hearn (1990) applied to the semi-asymmetric TAP, which modifies the network for the subproblem and has part of the network in the master problem, GSD-VI applied to the semi-asymmetric TAP produces an NLP subproblem that is almost the same as the whole model (in particular, the whole model’s constraint set), and the master problem has none of the network. If a symmetric TAP is to be improved by introducing some asymmetric costs, then this makes it relatively easy to code GSD-VI by using the original symmetric model, with minor modifications, as the subproblem. Another advantage of GSD-VI compared with the Benders method of Lawphongpanich and Hearn (1990) is that the master problem of GSD-VI is an ordinary VI, but in Lawphongpanich and Hearn (1990) it is in a class of generalized VI problems which is harder to solve; they proposed an untested method developed by Fang and Peterson (1982) for solving this generalized VI problem.

Potential benefits of decomposition for optimization or VI problems include: (1) reduction in memory requirements when the subproblem splits into several independent problems; (2) shorter computation times, e.g., if the subproblem is easy to solve and few iterations are required for convergence; and (3) easing of model development and management (Murphy, 1993), e.g., by allowing separate teams of developers to concentrate on several independent subproblems. The first is not relevant for GSD-VI, unless the subproblem splits into separate problems; this does not happen when applied to the TAP. The second advantage can sometimes occur for GSD-VI, as shown in some computational tests on TAPs in Section 4. There may be a model management advantage for GSD-VI
when applied to the TAP: if a more complicated TAP is to be developed from an existing symmetric TAP, by including some asymmetries, then, as mentioned above, the original symmetric model can be left largely intact as the subproblem, and the asymmetries can be introduced only in the master problem, which has a simple structure for the constraint set.

In short, the contributions of this paper are:

1. to derive the GSD-VI algorithm as a special case of DW decomposition for VIs;

2. to show that LH-SD of VI (Lawphongpanich and Hearn, 1984) is a special case of GSD-VI;

3. to show that the assumptions for the convergence proofs of Lawphongpanich and Hearn (1984) and García et al. (2003) can be relaxed somewhat by relying on the convergence theory of DW decomposition for VIs;

4. to demonstrate GSD-VI on several semi-asymmetric TAPs with varying numbers of asymmetric arcs, with comparisons to solving without decomposition;

5. to compare GSD-VI to LH-SD on several asymmetric TAPs; and

6. to study the effectiveness of the column dropping technique and the master problem approximation of the LH-SD method.

Section 2 reviews DW decomposition for VIs in Fuller and Chung (2005) and Gabriel and Fuller (2010), and LH-SD of VI for the asymmetric traffic assignment problem in
Lawphongpanich and Hearn (1984). Section 3 describes the general method by which we can employ DW decomposition for VIs that have difficult and easy variables, with an application to the semi-asymmetric VI. Section 3 also shows that LH-SD of VI is a special case of GSD-VI. Section 4 presents computational results using several TAPs, comparing GSD-VI to no decomposition, GSD-VI to LH-SD of VI, and LH-SD for VI with and without column dropping and master problem approximation. Section 5 presents conclusions.

2 Summary of DW and LH-SD Methods for VI

The class of VI that we study may be described as follows. Throughout this paper, all vectors are column vectors and the symbol ‘$T$’ as a superscript indicates the transpose of a vector or matrix. Given a set $S \subseteq \mathbb{R}^n$ and a mapping $C : S \rightarrow \mathbb{R}^n$, the VI problem is to find an $x^* \in S$ such that $C^T(x^*)(x - x^*) \geq 0$ for all $x \in S$. For example, in the user equilibrium traffic assignment problem $S$ is a closed convex set of feasible flow patterns, $n$ is the number of arcs in the network, and $C(x)$ is the arc delay vector (Lawphongpanich and Hearn, 1984).

In order to provide a summary of the DW and LH-SD methods for VIs, we consider a VI defined over the set $S = \{x \in \mathbb{R}^n | g(x) \geq 0, h(x) \geq 0\}$, where the vector-valued functions $g$ and $h$ have entries which are concave (making $S$ convex) and continuously differentiable. The complicating constraints $h(x) \geq 0$ make the VI difficult: they are dropped from the formulation of the subproblem, as in Dantzig-Wolfe decomposition of a
constrained optimization problem. In order to derive the method in Section 3, we need to
distinguish two types of variables, \( x_1 \in \mathbb{R}^{n_1}, x_2 \in \mathbb{R}^{n_2}, \) with \( n = n_1 + n_2, \) i.e., \( x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}, \)
which we write as \((x_1^T, x_2^T)^T\) to save space. The function \( g(x) \) depends only on \( x_2, \) and
we write \( g(x_2), \) while \( h(x) \) depends on both \( x_1 \) and \( x_2. \) In addition, \( h(x) \) is assumed to
be separable as \( h(x) = h_1(x_1) + h_2(x_2), \) and \( C(x) = (C_1^T(x_1), C_2^T(x_2))^T. \) Thus, the VI
problem that we consider is

\[
VI(S, C): \text{ With }
\]
\[
S = \{(x_1^T, x_2^T)^T \in \mathbb{R}^{n_1+n_2} | g(x_2) \geq 0, h_1(x_1) + h_2(x_2) \geq 0\} \tag{1}
\]
find \((x_1^*, x_2^*)^T \in S\) such that
\[
C_1^T(x_1^*) (x_1 - x_1^*) + C_2^T(x_2^*) (x_2 - x_2^*) \geq 0, \forall (x_1^T, x_2^T)^T \in S. \tag{2}
\]

2.1 Dantzig-Wolfe Decomposition of VI(S, C)

In the Dantzig-Wolfe decomposition procedure, the subproblem VI has only \( x_2 \) variables.
At iteration \( k, \) the subproblem drops the complicating constraints \( h_1(x_1) + h_2(x_2) \geq 0 \) and
introduces an adjustment to the mapping \( C_2, \) which depends on the dual variable vector
\( \beta^{k-1} \) associated with the master problem constraints \( h_1(x_1) + h_2(x_2) \geq 0, \) and on the
value \( x_{2M}^{k-1} \) calculated at the last iteration by the master problem VI. With these changes,
the subproblem becomes

**Sub-VI** \(^k\): With

\[ \mathcal{S} = \{ x_2 \in \mathbb{R}^{n_2} | g(x_2) \geq 0 \} \]  

find \( x_{2S}^k \in \mathcal{S} \) such that

\[ (C_2(x_{2S}) - \nabla h_2^T(x_{2M}^k)\beta^{k-1})^T(x_2 - x_{2S}^k) \geq 0, \ \forall x_2 \in \mathcal{S}. \]  

The definition of the master problem at iteration \( k \) relies on the \( n_2 \times k \) matrix whose columns are the solutions of the previous \( k \) subproblems (the ‘proposals’):

\[ X^k_2 = [x_{2S}^1, x_{2S}^2, \ldots, x_{2S}^k] \]  

The feasible set for the master problem is defined over \( x_1 \) vectors and weights on proposals as

\[ \Lambda^k = \{(x_1^T, \lambda^T)^T \in \mathbb{R}^{n_1+k} | h_1(x_1) + h_2(X_2^k \lambda) \geq 0, \ e^{kT} \lambda = 1, \ \lambda \geq 0 \} \]

where \( e^k \) is a column vector whose \( k \) entries are all ones. That is, \( \Lambda^k \) is the set of \( x_1 \) vectors and those convex combinations of the previous subproblem solutions that are feasible for the complicating constraints. The master problem VI at iteration \( k \) is given as follows:
Master-VI$^k$: With $\Lambda^k$ defined as in (6)

\[
\text{find } (x^{kT}_1, \lambda^{kT})^T \in \Lambda^k \text{ such that }
\]

\[
C^T_1(x^{k}_1)(x_1 - x^{k}_1) + C^T_2(X^k_2\lambda^k)X^k_2(\lambda - \lambda^k) \geq 0, \forall (x^T_1, \lambda^T)^T \in \Lambda^k . \tag{7}
\]

The solution $(x^{kT}_1, \lambda^{kT})^T$ may be expressed as $x^k_M = (x^{kT}_1, x^{kT}_2)^T \equiv (x^{kT}_1, (X^k_2\lambda^k)^T); x^{k}_2$ and the complicating constraints’ dual variables $\beta^k$ are passed to the next subproblem, Sub-VI$^{k+1}$.

Using the results of Fuller and Chung (2005) and Gabriel and Fuller (2010), we can define a scalar quantity called the convergence gap:

\[
\gamma^k = (C^T_2(x^{k-1}_2) - \nabla h^T_2(x^{k-1}_2)\beta^{k-1})^T(x^k_S - x^{k-1}_2) . \tag{8}
\]

Under very mild conditions (continuity of $C$, and existence of at least one limit point of $\{(x^k_{2M}, \beta^k, x^{k+1}_{2S})\}_{k=1}^{\infty}$), it is shown that (a) if $\gamma^k < 0$ then $x^k_{2S}$ cannot be written as a convex combination of the earlier proposals in $X^{k-1}_2$ (i.e. the algorithm is still making progress), and (b) either $\gamma^k \geq 0$ at a finite iteration number $k$, or $\gamma^k \rightarrow 0$ as $k \rightarrow \infty$.

Under stricter sufficient conditions, it is also shown that if $\gamma^k \geq 0$, then $(x^{k-1T}_1, x^{k-1T}_{2M})^T$ solves $VI(S,C)$. Two stricter conditions were identified, either one of which guarantees
convergence to a solution of $VI(S,C)$:

$$C_2 \text{ is strictly monotone, or}$$

$$C_2(x_2) = (F^T(q), \nabla d^T(y))^T, \quad (9a)$$

where $x_2 = (q^T, y^T)^T$, $q$ and $F$ have the same dimension,

$F$ is strictly monotone and $d$ is convex.

Condition (9b) includes the special case that $C_2(x_2) = \nabla d(x_2)$, which we use in Section 3.1, in the application to the semi-asymmetric TAP.

Based on these results, the algorithm is designed to stop when $\gamma_k > -\varepsilon$, where $\varepsilon > 0$ is a predetermined convergence tolerance. The algorithm is defined as follows. It is assumed that a reliable method is available to solve the master and subproblem VIs, e.g., a relaxation algorithm in Dafermos (1982), or Extended Mathematical Programming (EMP) in GAMS (Ferris et al., 2009) which reformulates the VI as a mixed complementarity problem and calls PATH as the solver.

**Dantzig-Wolfe Algorithm for $VI(S,C)$**

**Step 0:** Set $k = 0$. Choose $\varepsilon > 0$. Set $X_2^0$ to the null matrix, and choose a value for $\nabla h^T_2(x_{2M}^0)\beta^0$. Go to Step 1.

**Step 1:** Increment $k \leftarrow k + 1$. Solve Sub-VI$^k$ (3), (4). If $k = 1$ and the subproblem is infeasible, then STOP; else place the solution $x_{2S}^k$ in the matrix $X^k_2 = [X^{k-1}_2, x_{2S}^k]$.

If $k = 1$, then go to Step 2; else \{calculate $\gamma^k$ (8) and if $\gamma^k > -\varepsilon$, then STOP; else
Step 2: Solve Master-VI$^k$ (6), (7). Record $\nabla h_2^T(x_{2M}^k)_{\beta^k}$. Go to Step 1.

Although there is a distinction between variables that appear in the master problem but not the subproblem, $x_1$, and variables that do appear in the subproblem, $x_2$, the convergence results from Fuller and Chung (2005) are still valid, with slight modifications. The details of such modifications can be found in Gabriel and Fuller (2010).

2.2 LH-Simplicial Decomposition of $VI(S,C)$

Lawphongpanich and Hearn (1984) presented a type of simplicial decomposition for VIs in the context of solving asymmetric traffic assignment problems. Simplicial decomposition does not exploit any distinctions between types of constraints, nor between types of variables, i.e., we can write $x$ instead of $(x_1^T, x_2^T)^T$, and $S = \{x \in \mathbb{R}^n | g(x) \geq 0\}$ instead of (1). The subproblem in simplicial decomposition has the same feasible region as the original problem, and because the subproblem mapping is a constant vector (equal to the value of the mapping from the previous master problem solution), the subproblem is a constrained optimization problem. If the constraint functions $g$ are affine, then the subproblem is a linear program, as is the case for the traffic assignment problem. The master problem is over the convex hull of proposals from the subproblem, as in Dantzig-Wolfe decomposition, but there are no complicating constraints that are imposed in the master problem. The algorithm defined by Lawphongpanich and Hearn (1984) has two features included to enhance computational efficiency – columns can be dropped, and the
master problem is solved only approximately (but with greater accuracy in later iterations). Applying Lawphongpanich-Hearn simplicial decomposition to VI$(S, C)$, we have the following algorithm.

**LH–Simplicial Algorithm for VI$(S, C)$**

**Step 0:** Choose a convergent monotone sequence $\{\epsilon_k\}_{k=1}^{\infty}$, where $\epsilon_k > \epsilon_{k+1} > 0$, and $\epsilon_k \to 0$ as $k \to \infty$. Choose $\delta > 0$. Let $x^1_S$ be an extreme point of $S$, and define $X^1 = \{x^1_S\}$, $G^1 = \infty$ and $k = 1$.

**Step 1:** Solve the master variational problem:

$$\text{Find } x^k_M \in \text{conv}(X^k) \text{ such that } C^T(x^k_M)(x - x^k_M) \geq -\epsilon_k, \forall x \in \text{conv}(X^k)$$

(10)

where $\text{conv}(X^k)$ is the convex hull of $X^k$.

**Step 2:** Solve the subproblem:

$$\min\{C^T(x^k_M)x : x \in S\} = C^T(x^k_M)x^k_S.$$  

(11)

If $G(x^k_M) \triangleq C^T(x^k_M)(x^k_M - x^k_S) = 0$ STOP. Otherwise,

(i) if $G(x^k_M) \geq G^k - \delta$ let $X^{k+1}_M = X^k \cup x^k_S$;

(ii) if $G(x^k_M) < G^k - \delta$ let $X^{k+1} = (X^k \setminus D^k) \cup x^k_S$. 

13
Let $G^{k+1} = \min\{G^k, G(x_M^k)\}$ and increment $k$. Go to Step 1.

The parameter $\delta$ is a small positive tolerance for the column dropping decision, with the smaller the value of $\delta$, the more column dropping being executed. The set $D^k = \text{columns of } X^k \text{ with zero weight in the expression of } x_M^k \text{ as a convex combination of columns of } X^k$ in Step 1.

The parameters $\epsilon_k$ allow for the master problem to be solved approximately. The convergence proof of LH–Simplicial Algorithm for $VI(S, C)$ can be found in Lawphongpanich and Hearn (1984). Strong monotonicity of $C(x)$ is assumed, for convergence.

### 3 Generalized Simplicial Decomposition

Some transportation network equilibrium problems are semi-asymmetric, i.e., the variable vector $x = \begin{pmatrix} x_A \\ x_B \end{pmatrix} = (x_A^T, x_B^T)^T$, where the subvector $x_A$ has a non-integrable cost function $C_A(x_A)$, and $x_B$ has an integrable cost function $C_B(x_B)$. This is the motivation for the generalized simplicial decomposition algorithm described in this section, but the algorithm can be applied to any VI model of the following form, with ‘difficult’ variables $x_A$ and ‘easy’ variables $x_B$: 

---

14
\[ VI(S_{AB}, C_{AB}): \text{ With} \]
\[ S_{AB} = \{(x_A^T, x_B^T)^T \in \mathbb{R}^{n_A+n_B} \mid g(x_A, x_B) \geq 0\} \]  
(12)

find \((x_A^*, x_B^*)^T \in S_{AB}\) such that
\[ C_A^T(x_A^*)((x_A - x_A^*) + C_B^T(x_B^*)(x_B - x_B^*) \geq 0, \forall (x_A^T, x_B^T)^T \in S_{AB}. \]  
(13)

We derive the generalized simplicial decomposition algorithm in three steps: (i) we state an equivalent form of \(VI(S_{AB}, C_{AB})\), using a dummy variable vector \(z\) and constraints \(z - x_A = 0\); (ii) we apply the DW algorithm for VIs of Section 2.1, with complicating constraints \(z - x_A = 0\); and (iii) we simplify the resulting master problem and subproblem in part by eliminating \(z\).

For the first step, it is clear that the following VI problem is equivalent to \(VI(S_{AB}, C_{AB})\):

\[ VI(bS_{AB}, bC_{AB}): \text{ With} \]
\[ bS_{AB} = \{(z^T, x_A^T, x_B^T)^T \in \mathbb{R}^{2n_A+n_B} \mid g(x_A, x_B) \geq 0, z - x_A = 0\} \]
(14)

find \((z^*, x_A^*, x_B^*)^T \in bS_{AB}\) such that
\[ C_A^T(z^*)(z - z^*) + C_B^T(x_B^*)(x_B - x_B^*) \geq 0, \forall (z^T, x_A^T, x_B^T)^T \in bS_{AB}. \]  
(15)

For the second step, we can apply DW decomposition to \(VI(\hat{S}_{AB}, \hat{C}_{AB})\), with \(z\) as the vector of variables that appear only in the master problem, and \(z - x_A = 0\) as the complicating constraints. The correspondence with the symbols used in Section 2.1 is
indicated below by ‘∼’:

\[
\begin{align*}
    x_1 & \sim z \\
    C_1(x_1) & \sim C_A(z) \\
    x_2 & \sim (x_A^T, x_B^T) \\
    C_2(x_2) & \sim (0^T, C_B^T(x_B))^T \\
    h_1(x_1) + h_2(x_2) & \sim z - x_A.
\end{align*}
\]

With this structural interpretation, the Dantzig-Wolfe subproblem (3), (4), master problem (6), (7) and convergence gap (8) can be defined as follows.

**Sub-VI-GSD\(_{AB}^k\):** With \(S_{AB}\) as in (12)

\[
\text{find } (x_{AS}^k, x_{BS}^k)^T \in S_{AB} \text{ such that } \\
\beta_{k-1}(x_A - x_{AS}^k) + C_A^T(x_{BS}^k)(x_B - x_{BS}^k) \geq 0, \ \forall (x_A^T, x_B^T)^T \in S_{AB}
\]

(16)

**Master-VI-GSD\(_{AB}^k\):** With

\[
\tilde{S}_{ABM}^k = \{(z^T, \lambda^T)^T \in \mathbb{R}^{nA+k} | z - X_A^k \lambda^k = 0, e^k \lambda - 1 = 0, \lambda \geq 0\}
\]

(17)

find \((z^*, \lambda^*)^T \in \tilde{S}_{ABM}^k\) such that

\[
C_A^T(z^*) + C_B^T(X_B^k \lambda^*) X_B^k (\lambda - \lambda^*) \geq 0, \ \forall (z^T, \lambda^T)^T \in \tilde{S}_{ABM}^k.
\]

(18)
The convergence gap (8) becomes:

\[ \hat{\gamma}^k_{AB} = \beta^{k-1T}(x^k_{AS} - x^{k-1}_{AM}) + C^T_B(x^{k-1}_{BM})(x^k_{BS} - x^{k-1}_{BM}) \ . \tag{19} \]

The third and final step to derive the generalized simplicial decomposition algorithm is to replace \( z \) in \( \text{Master-VI-GSD}^k_{AB} \) by \( X^k_A\lambda^k \), and to replace \( \beta^{k-1} \) in \( \text{Sub-VI-GSD}^k_{AB} \) by the value of \( C_A \) that was calculated by the previous master problem solution. The replacement of \( \beta^{k-1} \) is justified by Karush-Kuhn-Tucker (KKT) conditions for the solution of \( \text{Master-VI-GSD}^k_{AB} \) shown in Theorem 1 below. Since the result is almost immediate from Proposition 2.2 of Harker and Pang (1990), no proof is provided here.

**Theorem 1.** \( \lambda^k, z^k \) solve \( \text{Master-VI-GSD}^k_{AB} \) (17), (18) iff there exist \( \lambda^k \in \mathbb{R}^n_+ \), \( z^k \in \mathbb{R}^{n_A} \), \( \beta^k \in \mathbb{R}^{n_A} \), and \( \theta^k \in \mathbb{R} \) such that all of the following conditions are satisfied:

\[
\begin{align*}
X_B^{kT}C_B(X_B^{k}\lambda^k) + X_A^{kT}\beta^k - e^k\theta^k &\geq 0 \quad (20a) \\
C_A(z^k) - \beta^k &= 0 \quad (20b) \\
e^{kT}\lambda^k &= 1 \quad (20c) \\
\lambda^{kT}(X_B^{kT}C_B(X_B^{k}\lambda^k) + X_A^{kT}\beta^k - e^k\theta^k) &= 0 \quad (20d) \\
z^k - X_A^{k}\lambda^k &= 0 \ . \quad (20e)
\end{align*}
\]

Equations (20b) and (20e) imply that \( C_A(X_A^{k-1}\lambda^{k-1}) = \beta^{k-1} \), and since \( x_{AM}^k = X_A^{k}\lambda^k \), we have \( C_A(x_{AM}^{k-1}) = \beta^{k-1} \). Therefore, \( \text{Sub-VI-GSD}^k_{AB} \) (12), (16), \( \text{Master-VI-GSD}^k_{AB} \) (17),
(18) and $\gamma_{AB}^k$ (19) can be restated as follows.

**Sub-VI-GSD$^k_{AB}$**: With $S_{AB}$ as in (12)

find $(x_{AS}^k, x_{BS}^k)^T \in S_{AB}$ such that

$$C_A^T(x_{AM}^{k-1})(x_A - x_{AS}^k) + C_B^T(x_{BS}^k)(x_B - x_{BS}^k) \geq 0, \forall (x_A^T, x_B^T)^T \in S_{AB}.$$  \hspace{1cm} (21)

**Master-VI-GSD$^k_{AB}$**: With

$$S_M^k = \{ \lambda \in \mathbb{R}^k | e^{kT}\lambda = 1, \lambda \geq 0 \}$$

find $\lambda^k \in S_M^k$ such that

$$C_A^T(X_A^k\lambda^k)X_A^k(\lambda - \lambda^k) + C_B^T(X_B^k\lambda^k)X_B^k(\lambda - \lambda^k) \geq 0, \forall \lambda \in S_M^k.$$ \hspace{1cm} (23)

The convergence gap (19) becomes:

$$\gamma_{AB}^k = C_A^T(x_{AM}^{k-1})(x_{AS}^k - x_{AM}^{k-1}) + C_B^T(x_{BM}^{k-1})(x_{BS}^k - x_{BM}^{k-1}).$$ \hspace{1cm} (24)

We can now state the generalized simplicial decomposition algorithm for $VI(S_{AB}, C_{AB})$.

**GSD-VI Algorithm**

**Step 0**: Set $k = 0$. Choose $x_{AM}^0$ and $\varepsilon > 0$. Set $X_A^0, X_B^0$ to the null matrix. Go to Step 1.

**Step 1**: Increment $k \leftarrow k + 1$. Solve Sub-VI-GSD$^k_{AB}$ (12), (21). If $k = 1$ and the
subproblem is infeasible, then STOP; else place the solution in the matrices $X^k_A = [X_{A}^{k-1}, x^{k}_{AS}]$, $X^k_B = [X_{B}^{k-1}, x^{k}_{BS}]$.

If $k = 1$, then go to Step 2; else

{Calculate $\gamma^k_{AB}$ (24) and if $\gamma^k_{AB} > -\varepsilon$, then STOP; else go to Step 2}.

**Step 2:** Solve Master-VI-GSD$^k_{AB}$ (22), (23). Record $x^k_{AM}$ and $x^k_{BM}$. Go to Step 1.

Since the GSD-VI algorithm is derived as a special application of the Dantzig-Wolfe algorithm for VIs, all of the convergence theory of Fuller and Chung (2005) and Gabriel and Fuller (2010) is valid. In particular, the proof of convergence relies on either of the two assumptions (9) on $C_2(x) = (0^T, C_B^T(x_B))^T$, both of which are less restrictive than strong monotonicity required by Lawphongpanich and Hearn (1984) and García et al. (2003).

### 3.1 Application of GSD-VI Algorithm to Semi-asymmetric VI

In $VI(S_{AB}, C_{AB})$ (12), (13), if the mapping $C_A(x_A)$ is asymmetric, and the mapping $C_B(x_B)$ is symmetric, then $VI(S_{AB}, C_{AB})$ is said to be semi-asymmetric. For this structure, the subproblem, Sub-VI-GSD$^k_{AB}$, can be formulated as a nonlinear program because with $C_A(x^{k-1}_{AM})$ a constant vector and $C_B(x_B)$ integrable, the whole mapping is integrable. This may be computationally advantageous, compared with coding Sub-VI-GSD$^k_{AB}$ as a VI problem and calling a VI solver.

Convergence is guaranteed under assumption (9b) on $C_2(x) = (0^T, C_B^T(x_B))^T$, which here amounts to assuming that the integral of $C_B$ is a convex function. Note that we
do not need to assume any type of monotonicity for $C_A$. This is a considerably looser assumption than the strong monotonicity of $C = (C_A^T, C_B^T)^T$ required by García et al. (2003).

### 3.2 Simplicial Decomposition as a Special Case of GSD-VI Algorithm

In this subsection, we show that for $VI(S_{AB}, C_{AB})$ (12), (13), if $x = x_A$ – i.e., there is no subvector $x_B$ – then the subproblem and master problem have the same structures as the subproblem and master problem of simplicial decomposition as in Section 2.2. For this case, $x = x_A$, here are the subproblem and master problem for the GSD-VI algorithm:

**Sub-VI-SD$:**

With

$$S = \{x \in \mathbb{R}^n \mid g(x) \geq 0\}$$ (25)

find $x^k_S \in S$ such that

$$C^T(x^{k-1}_M)(x - x^k_S) \geq 0, \forall x \in S.$$ (26)
which is equivalent to the optimization subproblem (11) in the LH-Simplicial algorithm, because $C_A(x_{AM}^{k-1})$ is a constant vector; and

**Master-VI-SD**\(^k\): With $S^k_M$ as in (22)

find $\lambda^k \in S^k_M$ such that

$$C^T(X^k\lambda^k)X^k(\lambda - \lambda^k) \geq 0, \forall \lambda \in S^k_M.$$ 

which is the same as the master problem (10) in the LH-Simplicial algorithm.

The convergence gap is

$$\gamma^k = C^T(x_{M}^{k-1})(x_S^k - x_{M}^{k-1})$$

which is the negative of the stopping condition quantity $G(x^k_M)$ calculated at step 2 of the LH-Simplicial algorithm of Section 2.2. If the column dropping procedure and the master problem approximation were adopted in GSD-VI, then the two algorithms would be identical. Therefore, we can say that simplicial decomposition of Section 2.2 is a special case of the GSD-VI algorithm, and so the term ‘generalized’ is appropriate.

Applying the assumption (9b) to $C_2(x) = (0^T, C_B^T(x_B))^T = 0$ (because $x = x_A$ and there is no $C(x_B)$) leads to the ‘requirement’ that the vector 0 is the gradient of a convex function, which is true because $d(x) = 0$ is convex. In other words, there is no monotonicity requirement at all, for convergence, which is considerably looser than the strong monotonicity required by Lawphongpanich and Hearn (1984).
3.3 Remark

In Section 4, we report on the application of the GSD-VI algorithm to several semi-asymmetric traffic assignment models. Some calculations employ an NLP subproblem as described in Section 3.1 above, and other calculations use the special case of simplicial decomposition, i.e., with an LP subproblem as in Section 3.2. For the sake of having brief names for the two versions of the algorithm, we refer to the former as ‘GSD-VI Decomposition’ and the latter as ‘Simplicial Decomposition.’

4 Test Models and Computational Results

In our implementation, the test models, the decomposition algorithms and a reference algorithm without decomposition are coded into GAMS programs (Brooke et al., 1992), executed on a PC with Intel Core 2 Duo Processor and 4GB memory. The reference algorithm is a relaxation algorithm (Dafermos, 1982) by which the asymmetric test models are solved so that we can have reference results for evaluating the accuracy and speed of the decomposition algorithms. The master problem is also solved by the same relaxation algorithm. Each iteration of the relaxation algorithm is a nonlinear programming (NLP) calculation.

For all models solved by GSD-VI decomposition, in the first subproblem (12), (21) at step $k = 1$, the initial value of $C_A(x_{AM}^{k-1})$ is arbitrarily set to zero, which produces first proposals that ignore costs on asymmetric arcs. In the relaxation method for the reference calculations, the first NLP also has $C_A(x_{AM}^{k-1}) = 0$. For all models solved by Simplicial
Decomposition, the first subproblem (25), (26) the initial value of $C(x_M^{k-1})$ is arbitrarily set equal to 50 in every component of the vector. For all decomposition calculations, we set the convergence tolerance $\varepsilon = 0.00001$, except where noted. The relaxation iterations of the restricted master problems and the reference method, and all subproblems are solved by CONOPT 3 called from GAMS. In the calculations, many similar optimization problems are solved repeatedly – the subproblem, a sequence of optimization problems related to the restricted master problem, and another related to the reference algorithm; except for the first time, each time an optimization calculation is done, it is started from the last solution in order to reduce computation time.

If a symmetric TAP has already been developed, as an optimization problem, but it is now recognized that some of the arcs are better represented with asymmetric cost functions, then decomposition provides a development route to the semi-asymmetric model. The original, symmetric TAP can be used with only minor changes as the subproblem in either GSD-VI Decomposition, or Simplicial Decomposition. For both types of decomposition, the subproblem is an optimization problem, and the feasible set is given by (12) or equivalently (25), where $g(x_A, x_B) \geq 0$ or $g(x) \geq 0$ represents the affine constraints of the TAP. The minor changes to the original symmetric TAP are to make the ‘$A$’ terms in the objective function linear (for the subproblem of GSD-VI Decomposition), or to make all terms in the objective function linear (for Simplicial Decomposition).

The master problem for either type of decomposition has a simple feasible set (22) that is easy to code, and a method such as a relaxation algorithm (Dafermos, 1982) would have to be coded; an alternative, easier approach is to code the master problem directly.
as a VI, using EMP in GAMS (Ferris et al., 2009) to reformulate the master problem as a mixed complementarity problem and call PATH as the solver.

To summarize, even if a decomposition algorithm takes more time than no decomposition, there can still be an advantage to a decomposition approach, in model development. However, if decomposition takes a very large amount of time, then such an advantage might not be worthwhile. Therefore, we are interested in tests that measure the time of decomposition, compared with no decomposition, and in variants of decomposition algorithms that may run in shorter times, e.g., with column dropping or master problem approximations, as in Section 2.2.

4.1 GSD-VI Decomposition for Semi-asymmetric Models

In this subsection, we present information on the convergence behavior of GSD-VI Decomposition for several semi-asymmetric traffic assignment models, i.e., using the algorithm as described in Section 3.1, with an NLP subproblem.

Lawphongpanich and Hearn (1990) devised a Benders decomposition algorithm for VIs, which, when applied to the semi-asymmetric traffic assignment problem, had a symmetric subproblem and an asymmetric master problem (which accumulated cuts instead of proposals). Before we present our main illustrative examples, the test model of Lawphongpanich and Hearn (1990) is used to compare our GSD-VI algorithm with the one in Lawphongpanich and Hearn (1990). The computational result is encouraging, in that the number of decomposition steps of our decomposition method, 7, is much less than that of Lawphongpanich and Hearn, 50. However, we have not coded the Lawphong-
panich and Hearn (1990) Benders algorithm, so CPU times cannot be compared. Such a
time comparison would not be very informative about timing for most problems, because
the Lawphongpanich and Hearn (1990) subproblem has a special structure which allows
closed form solutions to be derived; most models do not admit closed form solutions.

4.1.1 Results from Sioux-Falls model (SF)

We first discuss how we build several semi-asymmetric equilibrium problems, based on
a symmetric traffic assignment problem (TAP) taken from the Sioux-Falls network ex-
ample of Ferris et al. (1999). Then, we present some computational results for GSD-VI
Decomposition and the reference algorithm.

The Sioux-Falls model has the network topology shown in Figure 1. There are 76
arcs (38 in Figure 1, times 2 directions for each), and for each arc $a$, we have the cost
functions $c_a(f_a) = u_a + b_a \times \left(\frac{f_a}{k_a}\right)^4$ where $u_a$, $b_a$, and $k_a$ are constants, and $f_a$ is the
traffic flow on arc $a$. Arcs are specified by their two nodes; e.g., $a = 1.2$ is the arc from
node 1 to node 2. All of these constants and the trip table can be obtained from the
Appendix of Ferris et al. (1999). We suppose that further investigation indicates that the
cost functions of 12 arcs (bold lines in Figure 1) are required to change to $c_a(f_a, f_{\tilde{a}}) =
\left(\frac{f_a + v_a \times f_{\tilde{a}}}{k_a}\right)^4$ where $v_a$ is an additional constant. The model becomes a semi-
asymmetric model called SF(12) with 12 indicating there are twelve asymmetric arcs in
the model. The changed cost functions are: $c_{1.2}(f_{1.2}, f_{1.3}), c_{1.3}(f_{1.3}, f_{1.2}), c_{3.12}(f_{3.12}, f_{3.4}),
c_{3.4}(f_{3.4}, f_{3.12}), c_{21.22}(f_{21.22}, f_{21.24}), c_{21.24}(f_{21.24}, f_{21.22}), c_{21.2}(f_{21.2}, f_{3.1}), c_{3.1}(f_{3.1}, f_{2.1}), c_{12.3}(f_{12.3}, f_{4.3}),
c_{4.3}(f_{4.3}, f_{12.3}), c_{22.21}(f_{22.21}, f_{24.21}), c_{24.21}(f_{24.21}, f_{22.21})$, and the corresponding $v_a$ are 0.25,
Arcs with asymmetric cost functions

Figure 1: The network of Sioux-Falls model

0.3, 0.1, 0.25, 0.25, 0.36, 0.25, 0.3, 0.1, 0.25, 0.25, and 0.36 respectively. Similar examples of asymmetric cost functions can be found in Lawphongpanich and Hearn (1984). Following GSD-VI Decomposition, we construct the corresponding master problem and modify the original symmetric TAP for the subproblem.

Computational results show that the maximum difference between the solutions \( f_a \) of the decomposition method and the reference method is 0.00005\%. Table 1 shows the value of the convergence gap \( \gamma_{AB}^k \) at each decomposition step of the algorithm. The raw convergence gap, a negative number, is in column two. Column three shows \( \log(1 - \gamma_{AB}^k) \), which is also shown in Figure 2 in a graph versus iteration number. The fourth column of Table 1 expresses the magnitude of the convergence gap (the improvement in total travel
Figure 2: Convergence gap versus iteration number for SF(12).

time) as a percentage of the equilibrium travel time.

| Decomposition Step, $k$ | Convergence Gap, $\gamma^k_{AB}$ | $\log(1-\gamma^k_{AB})$ | $|\gamma^k_{AB}|$ | Equilibrium travel time* (%) |
|------------------------|----------------------------------|--------------------------|----------------|-----------------------------|
| 1                      | -16601.900                       | 4.220183952              | 310.6733073    |                             |
| 2                      | -464.711                         | 2.668116496              | 8.6961916      |                             |
| 3                      | -35.135                          | 1.557928059              | 0.657485387    |                             |
| 4                      | -9.697                           | 1.029261996              | 0.181461102    |                             |
| 5                      | -0.589                           | 0.201123897              | 0.011022026    |                             |
| 6                      | -0.146                           | 0.059184618              | 0.002732115    |                             |
| 7                      | -0.001                           | 0.000434077              | 1.87131E-05    |                             |
| 8                      | -1.25813E-5                      | 5.46395E-06              | 2.35435E-07    |                             |
| 9                      | -1.44288E-6                      | 6.26634E-07              | 2.70008E-08    |                             |

*Equilibrium travel time = 5343.845

Table 1. Progress of the iterations for model SF(12), with $\varepsilon = 0.00001$. 
The convergence gap in Figure 2 drops rapidly (and monotonically, in this case, even though the convergence theory does not require monotonic improvement in the convergence gap). This rapid convergence may be because the number of asymmetric arcs is small, which implies a small number of linking constraints in the intermediate model used in the derivation of GSD-VI Decomposition, $VI(\tilde{S}_{AB}, \tilde{C}_{AB})$ (normally considered advantageous for Dantzig-Wolfe decomposition of optimization models). Another explanation for the rapid convergence is that, with a small number of asymmetric arcs, the subproblem is very similar to the original problem, so the subproblem perhaps delivers very good proposals to the master problem.

In order to show the behavior of the convergence gap against the number of asymmetric arcs, an additional four models are built by changing more symmetric arcs to asymmetric arcs arbitrarily in the model SF(12). These models are denoted SF(16), SF(20), SF(24) and SF(28), where again the numbers in brackets are the numbers of asymmetric arcs in the corresponding models. All of these models have the same network topology and trip matrix as the model SF(12), but the number of asymmetric arcs differs among them.

Table 2 summarizes the model descriptions. Table 3 reports the number of GSD-VI Decomposition steps required to achieve the very tight accuracy $\varepsilon = 0.00001$, and the computation times by the solver CONOPT3 for the master and subproblems, and for the reference method. As expected, the number of required decomposition steps and computation time increase with the number of asymmetric arcs. The reference method, without decomposition, takes more time than the decomposition method for the models with a smaller number of asymmetric arcs. However, as the number of asymmetric arcs
increases, the reference method is faster than the decomposition method. Figure 3 shows the decreasing pattern of the convergence gaps with increasing number of asymmetric arcs. All models indicate that the convergence gap approaches zero non-monotonically (except SF(12)) as discussed in Chung et al. (2006) and Fuller and Chung (2005). In addition, the more the number of asymmetric arcs in the model, the slower the convergence gap approaches zero.

<table>
<thead>
<tr>
<th>Model Name</th>
<th>Number of asymmetric arcs with descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>SF(12)</td>
<td>12 (see text for description)</td>
</tr>
<tr>
<td>SF(16)</td>
<td>16: asymmetric arcs in SF(12) and</td>
</tr>
<tr>
<td></td>
<td>$c_{13.24}(f_{13.24}, f_{12.13})$ with $v_{13.24} = 0.5$,</td>
</tr>
<tr>
<td></td>
<td>$c_{24.13}(f_{24.13}, f_{13.12})$ with $v_{24.13} = 0.3$,</td>
</tr>
<tr>
<td></td>
<td>$c_{12.13}(f_{12.13}, f_{13.14})$ with $v_{12.13} = 0.76$,</td>
</tr>
<tr>
<td></td>
<td>$c_{13.12}(f_{13.12}, f_{24.13})$ with $v_{13.12} = 0.36$,</td>
</tr>
<tr>
<td>SF(20)</td>
<td>20: asymmetric arcs in SF(16) and</td>
</tr>
<tr>
<td></td>
<td>$c_{2.6}(f_{2.6}, f_{8.6})$ with $v_{2.6} = 0.86$,</td>
</tr>
<tr>
<td></td>
<td>$c_{6.2}(f_{6.2}, f_{6.6})$ with $v_{6.2} = 0.46$,</td>
</tr>
<tr>
<td></td>
<td>$c_{6.8}(f_{6.8}, f_{2.6})$ with $v_{6.8} = 0.56$,</td>
</tr>
<tr>
<td></td>
<td>$c_{8.6}(f_{8.6}, f_{6.2})$ with $v_{8.6} = 0.36$,</td>
</tr>
<tr>
<td>SF(24)</td>
<td>24: asymmetric arcs in SF(20) and</td>
</tr>
<tr>
<td></td>
<td>$c_{7.8}(f_{7.8}, f_{8.9})$ with $v_{7.8} = 0.16$,</td>
</tr>
<tr>
<td></td>
<td>$c_{8.7}(f_{8.7}, f_{9.8})$ with $v_{8.7} = 0.86$,</td>
</tr>
<tr>
<td></td>
<td>$c_{8.9}(f_{8.9}, f_{7.8})$ with $v_{8.9} = 0.26$,</td>
</tr>
<tr>
<td></td>
<td>$c_{9.8}(f_{9.8}, f_{8.7})$ with $v_{9.8} = 0.76$,</td>
</tr>
<tr>
<td>SF(28)</td>
<td>28: asymmetric arcs in SF(24) and</td>
</tr>
<tr>
<td></td>
<td>$c_{10.16}(f_{10.16}, f_{17.16})$ with $v_{10.16} = 0.5$,</td>
</tr>
<tr>
<td></td>
<td>$c_{16.10}(f_{16.10}, f_{16.17})$ with $v_{16.10} = 0.3$,</td>
</tr>
<tr>
<td></td>
<td>$c_{16.17}(f_{16.17}, f_{10.16})$ with $v_{16.17} = 0.76$,</td>
</tr>
<tr>
<td></td>
<td>$c_{17.16}(f_{17.16}, f_{16.10})$ with $v_{17.16} = 0.36$,</td>
</tr>
</tbody>
</table>

Table 2. Descriptions of SF models.
Figure 3: Convergence gaps for all SF models and $\varepsilon = 0.00001$.

<table>
<thead>
<tr>
<th>Model Name</th>
<th>Decomposition steps ($k$) for $\varepsilon = 0.00001$</th>
<th>Decomposition steps ($k$) for $\varepsilon = 0.1$</th>
<th>Time for ref. method</th>
<th>Time for decomposition method</th>
</tr>
</thead>
<tbody>
<tr>
<td>SF(12)</td>
<td>8</td>
<td>7</td>
<td>1.047 sec.</td>
<td>0.938 sec.</td>
</tr>
<tr>
<td>SF(16)</td>
<td>13</td>
<td>10</td>
<td>1.672 sec.</td>
<td>1.531 sec.</td>
</tr>
<tr>
<td>SF(20)</td>
<td>14</td>
<td>12</td>
<td>1.750 sec.</td>
<td>1.750 sec.</td>
</tr>
<tr>
<td>SF(24)</td>
<td>20</td>
<td>15</td>
<td>2.797 sec.</td>
<td>2.515 sec.</td>
</tr>
<tr>
<td>SF(28)</td>
<td>29</td>
<td>22</td>
<td>3.313 sec.</td>
<td>3.562 sec.</td>
</tr>
</tbody>
</table>

Table 3. Number of decomposition steps and computation time (seconds).

However, the convergence tolerance of 0.00001 is extremely tight – in model SF(12), this corresponds to a travel time improvement (by the next subproblem solution) of $2.35 \times 10^{-5}\%$ compared with the equilibrium travel time; see Table 1. More reasonable, larger tolerances would stop the decomposition algorithm sooner; this is illustrated in Table 3, for a tolerance $\varepsilon = 0.1$, which is $1.87 \times 10^{-3}\%$ of the equilibrium travel time for
Finally, we note again that there can be value to a decomposition algorithm even if it does take somewhat longer than the nondecomposition approach, and for this set of tests, the only case for which decomposition took longer than the reference method — SF(28) with ε = 0.00001 — it took only 7.5% longer.

4.1.2 Results from Anaheim model

Since the decomposition algorithm is a little faster than the nondecomposition approach in SF models, it may suggest that the decomposition algorithm is actually faster than the non-decomposition approach for larger models with few asymmetric arcs. To empirically study this, a large scale real traffic assignment problem found at http://www.bgu.ac.il/~bargera/tntp is modified by converting a few symmetric arcs to asymmetric. The network is representative of the City of Anaheim, California, and consists of 416 nodes, 914 links and 38 origins and destinations. The cost function is the same form as the one in the Sioux-Falls illustration of Section 4.1.1. Table 4 provides some examples of the modified asymmetric arcs. Some models are denoted AN(XX) where the numbers in brackets are the numbers of asymmetric arcs in the corresponding models. For example, AN(20) means that there are 20 asymmetric functions. Other models, denoted AN(XX_XX), are highly asymmetric — e.g., AN(25_5) consists of 25 asymmetric cost functions, but for 5 of the asymmetric cost functions $c_a(f_a, f_{\tilde{a}})$, there is no asymmetric cost function $c_{\tilde{a}}(f_{\tilde{a}}, f_a)$ — the cost function for $\tilde{a}$ depends only on $f_{\tilde{a}}$, as $c_{\tilde{a}}(f_{\tilde{a}})$, or to put it another way, $v_{\tilde{a}} = 0$. For example, for $a = (10.388)$ in AN(25_5), there is no corresponding asymmetric function with $a =$
(388.10), but in AN(30), there is. The Jacobian matrix for an AN(XX_XX) model is highly asymmetric.

<table>
<thead>
<tr>
<th>Model Name</th>
<th>Number of asymmetric arcs with descriptions</th>
</tr>
</thead>
</table>
| AN(10_10) | $c_{1.117}(f_{1.117}, f_{1.117})$ with $v_{1.117} = 0.25$,
| | $c_{2.87}(f_{2.87}, f_{2.87})$ with $v_{2.87} = 0.25$,
| | $c_{3.74}(f_{3.74}, f_{74.73})$ with $v_{3.74} = 0.65$,
| | $c_{4.233}(f_{4.233}, f_{233.232})$ with $v_{4.233} = 0.30$,
| | $c_{5.165}(f_{5.165}, f_{165.164})$ with $v_{5.165} = 0.50$,
| | $c_{6.213}(f_{6.213}, f_{66.6})$ with $v_{6.213} = 0.80$,
| | $c_{7.253}(f_{7.253}, f_{253.252})$ with $v_{7.253} = 0.25$,
| | $c_{8.411}(f_{8.411}, f_{118.8})$ with $v_{8.411} = 0.25$,
| | $c_{9.379}(f_{9.379}, f_{379.9})$ with $v_{9.379} = 0.85$,
| | $c_{9.395}(f_{9.395}, f_{395.9})$ with $v_{9.395} = 0.75$, |
| AN(20) | Asymmetric arcs in AN(10_10) and
| | $c_{117.116}(f_{117.116}, f_{1.117})$ with $v_{117.116} = 0.75$,
| | $c_{87.86}(f_{87.86}, f_{2.87})$ with $v_{87.86} = 0.85$,
| | $c_{74.73}(f_{74.73}, f_{3.74})$ with $v_{74.73} = 0.25$,
| | $c_{233.232}(f_{233.232}, f_{4.233})$ with $v_{233.232} = 0.25$,
| | $c_{165.164}(f_{165.164}, f_{5.165})$ with $v_{165.164} = 0.80$,
| | $c_{166.6}(f_{166.6}, f_{6.213})$ with $v_{166.6} = 0.50$,
| | $c_{253.252}(f_{253.252}, f_{7.253})$ with $v_{253.252} = 0.30$,
| | $c_{411.8}(f_{411.8}, f_{8.411})$ with $v_{411.8} = 0.65$,
| | $c_{379.9}(f_{379.9}, f_{9.379})$ with $v_{379.9} = 0.25$,
| | $c_{395.9}(f_{395.9}, f_{9.395})$ with $v_{395.9} = 0.25$, |
| AN(25_5) | Asymmetric arcs in AN(20) and
| | $c_{10.338}(f_{10.338}, f_{338.10})$ with $v_{10.338} = 0.65$,
| | $c_{10.362}(f_{10.362}, f_{362.10})$ with $v_{10.362} = 0.30$,
| | $c_{11.309}(f_{11.309}, f_{309.11})$ with $v_{11.309} = 0.50$,
| | $c_{12.275}(f_{12.275}, f_{275.12})$ with $v_{12.275} = 0.80$,
| | $c_{13.262}(f_{13.262}, f_{262.13})$ with $v_{13.262} = 0.25$, |
| AN(30) | Asymmetric arcs in AN(25_5) and
| | $c_{338.10}(f_{338.10}, f_{10.338})$ with $v_{338.10} = 0.25$,
| | $c_{362.10}(f_{362.10}, f_{10.362})$ with $v_{362.10} = 0.85$,
| | $c_{309.11}(f_{309.11}, f_{11.309})$ with $v_{309.11} = 0.75$,
| | $c_{275.12}(f_{275.12}, f_{12.275})$ with $v_{275.12} = 0.75$,
| | $c_{262.13}(f_{262.13}, f_{13.262})$ with $v_{262.13} = 0.85$, |

Table 4. Descriptions of test models for Anaheim network.

Table 5 reports the computational results of AN(XX) models, where "% Asy" is the
% of asymmetric arcs for the model, "GSD-VI (Mas) Time" and "GSD-VI (Sub) Time" are the computation time of master and subproblem respectively by the solver CONOPT 3 from GAMS. "GSD-VI step" is the number of decomposition steps, "Ref. step" is the number of relaxation steps in the reference method, "GSD-VI Tot. Time" is the total solution time of decomposition, "Ref. Time" is the computation time of the reference method. The average elapsed time was about one hour. As expected, (1) the number of required decomposition steps and computation time usually increase with the number of asymmetric arcs; and (2) the reference method, without decomposition, takes more time than the decomposition method, for the models with smaller numbers of asymmetric arcs, in general. Changing the convergence tolerance from $\varepsilon = 0.00001$ to $\varepsilon = 0.1$ reduces the decomposition time for AN(60) and AN(100) to 3721.918 and 4513.597, respectively, closer to the reference times. The worst relative performance for GSD-VI was for the AN(100) case, which took 23.4% longer than the reference method.

<table>
<thead>
<tr>
<th>Model Name</th>
<th>% Asy</th>
<th>GSD-VI (Mas) Time (s.)</th>
<th>GSD-VI (Sub) Time (s.)</th>
<th>GSD Step</th>
<th>Ref. Step</th>
<th>GSD-VI Total Time (s.)</th>
<th>Ref. Time (s.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AN(20)</td>
<td>2.2%</td>
<td>0.484</td>
<td>2867.742</td>
<td>2</td>
<td>2</td>
<td>2868.226</td>
<td>3156.867</td>
</tr>
<tr>
<td>AN(30)</td>
<td>3.3%</td>
<td>0.527</td>
<td>3253.664</td>
<td>2</td>
<td>2</td>
<td>3254.191</td>
<td>3457.809</td>
</tr>
<tr>
<td>AN(40)</td>
<td>4.4%</td>
<td>0.578</td>
<td>3097.148</td>
<td>2</td>
<td>2</td>
<td>3097.726</td>
<td>3006.359</td>
</tr>
<tr>
<td>AN(50)</td>
<td>5.5%</td>
<td>0.613</td>
<td>3444.967</td>
<td>2</td>
<td>2</td>
<td>3445.580</td>
<td>3193.543</td>
</tr>
<tr>
<td>AN(60)</td>
<td>6.6%</td>
<td>0.516</td>
<td>4079.238</td>
<td>3</td>
<td>2</td>
<td>4079.754</td>
<td>3638.688</td>
</tr>
<tr>
<td>AN(100)</td>
<td>10.9%</td>
<td>0.813</td>
<td>4677.968</td>
<td>6</td>
<td>4</td>
<td>4678.781</td>
<td>3791.289</td>
</tr>
</tbody>
</table>

Table 5. Computational results of AN(XX) models for Anaheim network

The results of the highly asymmetric AN(XX_XX) models can be found in Table 6, which shows the performance of GSD-VI Decomposition. Solution times of AN(10_10),
AN(25_5), and AN(20_20) show that GSD-VI Decomposition performs better than the reference method, for small numbers of asymmetric arcs. The worst relative performance for GSD-VI was for the case AN(50_50), which took 28.9% longer than the reference method.

We suspect that GSD-VI Decomposition may sometimes be faster, in practice, than the non-decomposition approach, because some TAP models have only a few asymmetric functions and a highly asymmetric Jacobian matrix. For example, some urban areas may have a limited public transport service and the resulting percentage of asymmetric functions in the model is very small; see, e.g., de Cea et al. (2005).

<table>
<thead>
<tr>
<th>Model Name</th>
<th>% Asy</th>
<th>GSD-VI (Mas) Time (s.)</th>
<th>GSD-VI (Sub) Time (s.)</th>
<th>GSD Step</th>
<th>Ref. Step</th>
<th>GSD-VI Tot. Time (s.)</th>
<th>Ref. Time (s.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AN(10_10)</td>
<td>2.2%</td>
<td>0.617</td>
<td>3339.363</td>
<td>2</td>
<td>2</td>
<td>3339.980</td>
<td>4375.852</td>
</tr>
<tr>
<td>AN(25_5)</td>
<td>3.3%</td>
<td>0.562</td>
<td>2923.791</td>
<td>2</td>
<td>2</td>
<td>2924.353</td>
<td>3965.859</td>
</tr>
<tr>
<td>AN(20_20)</td>
<td>4.4%</td>
<td>0.547</td>
<td>3268.809</td>
<td>2</td>
<td>2</td>
<td>3269.356</td>
<td>3536.059</td>
</tr>
<tr>
<td>AN(30_30)</td>
<td>6.6%</td>
<td>0.500</td>
<td>3282.496</td>
<td>3</td>
<td>2</td>
<td>3282.996</td>
<td>3174.941</td>
</tr>
<tr>
<td>AN(50_50)</td>
<td>10.9%</td>
<td>0.734</td>
<td>4060.188</td>
<td>5</td>
<td>3</td>
<td>4060.922</td>
<td>3150.900</td>
</tr>
</tbody>
</table>

Table 6. Computational results of AN(XX_XX) models for Anaheim network

As expected, the decomposition algorithm is actually faster than the non-decomposition approach, for models with few asymmetric arcs. This empirical result can be considered as a counter-example of an argument of Patriksson (2004) that a decomposition over links is less often useful, as costs are not always defined by separable functions.
4.2 Comparing GSD-VI and Simplicial Decomposition for Some Semi-asymmetric Models

We also solved three of the Anaheim models using Simplicial Decomposition, i.e., the GSD-VI algorithm implemented as described in Section 3.2, with an LP subproblem. The subproblem has the structure of a shortest path problem (Lawphongpanich and Hearn, 1984), but since we do not have access to a specialized solver for such problems in the GAMS environment, we solved the subproblems with CONOPT. Therefore, our simplicial subproblem times are much longer than they could be with a shortest path algorithm, and direct comparisons of times with GSD-VI Decomposition or the reference are not useful.

Table 7 presents the number of decomposition steps, and master and subproblem CPU times, for Simplicial Decomposition, alongside the same information for GSD-VI Decomposition, and reference CPU times, from Tables 5 and 6. For two of the models, AN(20) and AN(100), Simplicial Decomposition took more than 60 steps to converge to the tolerance of \( \varepsilon = 0.00001 \), but due to limitations on the size of GAMS output files that we could open on our computer, we could not retrieve full information on times for these runs. For AN(20) and AN(100), Table 7 reports Simplicial Decomposition times for only 60 steps, for which the convergence gap \( \gamma^{60} \) achieved the values \(-0.06\) and \(-0.0001\), respectively.
<table>
<thead>
<tr>
<th>Model Name</th>
<th>GSD Step</th>
<th>GSD-VI (Mas) Time (s.)</th>
<th>GSD-VI (Sub) Time (s.)</th>
<th>Simp. Step</th>
<th>Simp. (Mas) Time (s.)</th>
<th>Simp. (Sub) Time (s.)</th>
<th>Ref. Time (s.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AN(20)</td>
<td>2</td>
<td>0.484</td>
<td>2867.742</td>
<td>over 60</td>
<td>6.211</td>
<td>4248.609</td>
<td>3156.867</td>
</tr>
<tr>
<td>AN(100)</td>
<td>6</td>
<td>0.813</td>
<td>4677.968</td>
<td>over 60</td>
<td>6.672</td>
<td>4127.719</td>
<td>3791.289</td>
</tr>
<tr>
<td>AN(50_50)</td>
<td>5</td>
<td>0.734</td>
<td>4060.188</td>
<td>22</td>
<td>6.109</td>
<td>4189.547</td>
<td>3150.900</td>
</tr>
</tbody>
</table>

Table 7. Computational results of GSD-VI vs. Simplicial for Anaheim network

Simplicial Decomposition requires a far greater number of steps than GSD-VI Decomposition for all three models. Apparently, the NLP subproblems of GSD-VI Decomposition provide much better proposals than the LP subproblems of Simplicial Decomposition, perhaps because the NLP subproblems are closer to the original model. García et al. (2003) suggest a related explanation for fewer decomposition steps with a nonlinear objective in the subproblem: an LP subproblem produces extreme point solutions, but an NLP subproblem (as an approximation of the original problem) can produce interior solutions. However, each LP subproblem for Simplicial Decomposition was solved by CONOPT much more quickly than each NLP subproblem of GSD-VI Decomposition, bringing the total computation times closer together than the number of steps. A shortest path algorithm would reduce the simplicial times even more, particularly if the different shortest path problems for different destinations were solved in parallel on separate processors.

### 4.3 Effectiveness of Column Dropping and Approximate Master in Simplicial Decomposition

As described in Section 2.2 above, Lawphongpanich and Hearn (1984) discussed the use of an approximate master problem for their simplicial decomposition algorithm, with the ap-
proximations improving at each step according to the user-specified convergent monotone sequence \( \{ \epsilon_k \} \). Intuitively, one might expect that such a strategy could save computation time in early iterations by requiring accuracy only near the end of the calculations.

In addition, Lawphongpanich and Hearn (1984) also discussed the importance of column dropping in real-world problem solving because it allows one to control the size of the master problem, which grows with each step in the number of variables (see Step 1 of LH-Simplicial Algorithm for \( VI(S,C) \) in Section 2.2 above). It is also expected that column dropping may improve the solution time of the master problem.

In this subsection, we study the effectiveness of the approximate master and the column dropping strategies, for Simplicial Decomposition.

4.3.1 Results from Sioux-Falls model

We begin with the Sioux-Falls model – a version in which all cost functions are asymmetric, SF(all) – solved by the simplicial decomposition algorithm of Section 2.2, for various \( \{ \epsilon_k \} \) sequences and column dropping tolerances \( \delta \). Similar to Lawphongpanich and Hearn (1984), the algorithms are terminated when the value of the relative gap

\[
\gamma^k = c^T(x_M^k)(x_S^{k+1} - x_M^k)/c^T(x_M^k)x_S^k \geq -0.000001.
\]

We do not use the value of \( -0.0001 \) used in Lawphongpanich and Hearn (1984) since we found that the algorithms stopped too early and could not provide the reference equilibrium solutions. In our tests, different combinations of \( \{ \epsilon_k \} \) and the column dropping parameter \( \delta \) (including no column dropping) are tested. The numbers of decomposition steps are presented in Table 8. The first row of results in Table 8 uses the values of \( \{ \epsilon_k \} \) from Lawphongpanich and Hearn (1984),
and the rightmost column uses the value of \( \delta \) from that paper. The second row uses a constant, small \( \epsilon_k \), i.e., the master problem is calculated to the same high degree of accuracy at each decomposition step. The remaining rows in Table 8 are for various nonconstant \( \{ \epsilon_k \} \) that differ in the accuracy required at any step \( k \). The maximum relative difference between the solution \( (f) \) obtained by the reference method and simplicial Decomposition is 0.001\% for all the tests. The average elapsed time was around two minutes.

<table>
<thead>
<tr>
<th>Convergent Sequence ( { \epsilon_k } )</th>
<th>Number of Decomposition Steps</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>No column dropping</td>
</tr>
<tr>
<td>( 0.1 + \frac{5 \times 0.0001}{k} )</td>
<td>141</td>
</tr>
<tr>
<td>( \frac{1000}{k} )</td>
<td>140 (reference)</td>
</tr>
<tr>
<td>( \frac{5 \times 0.001}{k} )</td>
<td>145</td>
</tr>
<tr>
<td>( \frac{5 \times 0.01}{k} )</td>
<td>141</td>
</tr>
<tr>
<td>( \frac{5 \times 0.1}{k} )</td>
<td>138</td>
</tr>
<tr>
<td>( \frac{5 \times 0.5}{k} )</td>
<td>145</td>
</tr>
</tbody>
</table>

*This \( \{ \epsilon_k \} \) and \( \delta = 0.0001 \) were used in Lawphongpanich and Hearn (1984).

Table 8. Results of simplicial decomposition steps for different \( \{ \epsilon_k \} \) and \( \delta \), SF(all)

Table 8 shows that if we use \( \{ \epsilon_k \} \) and \( \delta \) of Lawphongpanich and Hearn (1984), the number of decomposition steps decreases compared with no column dropping. Results of column "\( \delta = 0.01 \)" and "\( \delta = 0.0001 \)" imply that selection of the column dropping parameter \( \delta \) is critical. For most cases here, a smaller \( \delta \) (for more column dropping) is better. However, when \( \{ \epsilon_k \} = \frac{5 \times 0.01}{k} \) and \( \delta = 0.0001 \), column dropping does not help us to reduce the number of decomposition steps, and a smaller \( \delta \) is even worse. It is interesting that the number of steps is reduced substantially for most cases because of column dropping, but not because of the approximation of the master problem.
Table 9 reports the solution times, where "Master" and "Sub" are the summation, over all steps, of computation times of master problem and subproblem respectively. "Total" is the total solution time of Simplicial Decomposition. The approximation of the master problem slightly increases the computation times (compare row 2 to the other rows), and in particular, it increases the amount of time taken on master problem calculations. The column dropping procedure has a positive and more significant effect – it reduces the solution time of the master problem and the total solution time, except with \( \{\epsilon_k\} = \frac{5\times0.01}{k} \) and \( \delta = 0.0001 \). On the other hand, for columns with \( \delta = 0.01 \), although there is no reduction of decomposition steps, the solution time is reduced. Based on this test result, we find that the master approximation strategy does not help at all, but column dropping usually improves computational performance.

**4.3.2 Results from Arezzo model**

In order to further investigate whether simplicial decomposition with column dropping is actually faster than that without column dropping for large models, a large-scale real TAP
downloaded from the web site http://www2.ing.unipi.it/~d9762/research/test_networks.html of Mauro Passacantando is tested. The network of this TAP represents the extra-urban area of the city of Arezzo (Italy). It consists of 213 nodes, 598 arcs and 2423 O/D pairs. The form of the arc cost functions is the same as in the Sioux-Falls models. In the Arezzo model, all the asymmetric parameters $v_a = 0.5$. We used $\delta = 0.0001$ for column dropping because this worked well for the Sioux-Falls model.

<table>
<thead>
<tr>
<th>Convergent Sequence</th>
<th>Solution time (second)</th>
<th>No column dropping</th>
<th>$\delta = 0.0001$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>D_step</td>
<td>Tot</td>
<td>%_Diff</td>
</tr>
<tr>
<td>${0.1 + \frac{5+0.0001}{k}}$</td>
<td>57</td>
<td>946.414</td>
<td>0.03</td>
</tr>
<tr>
<td>${0.1 + \frac{5}{k}}$</td>
<td>58</td>
<td>969.617</td>
<td>0.05</td>
</tr>
<tr>
<td>${5+0.0001}$</td>
<td>58</td>
<td>962.383</td>
<td>0.08</td>
</tr>
<tr>
<td>${5+0.001}$</td>
<td>58</td>
<td>959.672</td>
<td>0.09</td>
</tr>
<tr>
<td>${5+1}$</td>
<td>54</td>
<td>942.734</td>
<td>0.05</td>
</tr>
</tbody>
</table>

Table 10. Computational results for Arezzo model.

Table 10 presents the computational results for the Arezzo model, where "D_step" is the number of decomposition steps, "Tot" is the total solution time. "%_Diff" is the maximum relative difference between the solution ($f$) obtained by the reference method and simplicial decomposition. The average solution time was around 16 minutes. Unexpectedly, the column dropping procedure increased the time for half of the $\{\epsilon_k\}$ sequences, and reduced the time only slightly for the other sequences.

Different values of $\delta$ produce different results. For example, for the $\{\epsilon_k\}$ of the first row in Table 10, increasing $\delta$ to 0.001 (from 0.0001) again requires 57 decomposition

\footnote{It was downloaded in 2007, but the website no longer exists.}
steps, but for slightly less time (935.008 seconds) than for no column dropping. An additional test shows that decreasing $\delta$ to 0.00001 requires more steps (63) and more time (1015.746 seconds) than for no column dropping. With such inconsistent and hard to predict results, we conclude that column dropping is not very useful for the TAPs tested.

5 Conclusions and Directions for Future Research

We have derived a generalized form of simplicial decomposition for VIs, called GSD-VI, from Dantzig-Wolfe decomposition for VIs, using dummy variables and ‘complicating’ constraints that equate the dummy variables to corresponding original model variables. Applied to a semi-asymmetric VI, the GSD-VI subproblem is an NLP approximation of the original model, of a different type than those discussed by García et al. (2003), and GSD-VI requires looser monotonicity conditions to prove convergence. When applied to the semi-asymmetric TAP, we need only assume that the symmetric part of the travel cost functions is monotone, i.e., that it is the gradient of a convex function. Another special case of GSD-VI produces essentially the same simplicial decomposition method as Lawphongpanich and Hearn (1984) for the asymmetric TAP, but with no monotonicity assumptions needed to prove convergence.

The Benders decomposition method of Lawphongpanich and Hearn (1990), applied to the semi-asymmetric TAP, also produces an NLP subproblem involving the symmetric arcs, but the master problem is a generalized VI, and the network is split between the master and subproblems. GSD-VI produces a normal VI master problem, and the entire
network is in the subproblem alone. We also show that the number of decomposition steps of GSD-VI is much less than that Benders decomposition on an example from Lawphongpanich and Hearn (1990).

Tests on several semi-asymmetric TAPs show that GSD-VI solves in less time than a reference method without decomposition when there are few asymmetric arcs in the model, and the worst performance observed, for models with many asymmetric arcs, had GSD-VI taking only 28.9% longer than the reference method.

Further tests on semi-asymmetric TAPs confirmed the expectation that GSD-VI, with the NLP subproblem, takes far fewer decomposition steps than simplicial decomposition (with an LP subproblem). However, the LP subproblem takes much less time to solve than the NLP subproblem, even with a general purpose LP solver. In future research, the solution time of the LP subproblem could possibly be shortened considerably if it can be solved as several shortest path problems in parallel, or by some recent efficient methods, like Bar-Gera (2002, 2006) or Babonneau and Vial (2008).

Other tests suggest that two features of the Lawphongpanich and Hearn (1984) algorithm – master problem approximation that is forced to be more accurate in later decomposition steps, and column dropping – may be ineffective (master approximation) or inconsistent (column dropping).

The reasonable computing times of GSD-VI and simplicial decomposition suggest that GSD-VI could be a useful development route to introduce asymmetries into an already existing symmetric TAP, using the existing TAP as the subproblem, with only minor modifications, and coding a master problem of a simple form. That is, decomposition can
be used to help to modify the traditional four-steps development procedure to develop a combined model. Since the traditional four-steps procedures with feedback may not have consistent results (see Siegel et al 2006), future research may examine the utility of decomposition in this way.

6 Acknowledgments

The work described in this paper was fully supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CityU 113008).

7 References


