Auction 알고리즘을 이용한 최단경로에 관한 연구

A Study on the Shortest path of use Auction Algorithm

₀우경환* · 이재영** · 이천희***
중경공업전문대* · Technical University of Budapest** · 청주대학교***

Woo, Kyong-Whan* · Yi, Jae-young** · Yi, Cheon-Hee***
Joongkyong Technical College* · Technical University of Budapest**
Chongju University***

Tel : 0431-229-8448 Fax : 0431-55-6392
E-mail : yicheon@soback.kornet.nm.kr

요 약

The classical algorithm for solving liner network flow problems are primal cost improvement method, including simplex method, which iteratively improve the primal cost by moving flow around simple cycles, which iteratively improve the dual cost by changing the prices of a subset of nodes by equal amounts. Typical iteration/shortest path algorithm is used to improve flow problem of liner network structure.

In this paper we stdudied about the implemental method of shortest path which is a practical computational aspects. This method can minimize the best neighbor node and also implement the typical iteration which is ε-CS satisfaction using the auction algorithm of linear network flow problem

1. 서론

선형망 흐름 문제를 해결하기 위한 전통적인 알고리즘은 단순한 방법을 포함하여 원시 비용 개선 방법이고, 이것은 단순 싸이클 흐름을 이동시킴으로서 원시 비용을 반복적으로 개선하고, 동일한 양으로 노드의 부분 집합 값을 변환하여줌으로서 이중 비용을 반복적으로 개선하는 것인데 이것을 이중 상승(dual ascent)방법이라 한다. 본 논문의 주제인 auction 알고리즘은 다른 알고리즘 보다는 빠르며, 선형망 구조에서 흐름 문제를 개선하기 위해 전형적인 반복/최단 경로 알고리즘을 이용할 수 있다. 비록 마지막에 그것들이 최적의 원시적인 해법을 찾는다 하더라도, 한 번의 반복으로 그것들은 원시 비용과 이중 비용을 저하시킬 수 있다. Bertsekas와 Mitter이[1] 제안한 Action 알고리즘은 비차별적 최적화(nondifferentiable optimization)와 ε-서브 구배법(ε-subgradient method)을 사용하였다.

Auction 알고리즘은 매우 직관적이고 이해하기 쉬우며, economic competetion concepts(경제적 경쟁개념)으로 설명될 수 있다. Action 알고리즘은 Bertsekas와 Eckstein의[2 ,3, 4]에 의해서 좀 더 발전되었으며, 차후 연구에서는 선형망 흐름 문제로까지 확장되어야 한다.

본 논문의 목적은 auction 알고리즘에 대한 개별적 소개를 제공하며[5, 6] 할당 문제와 최단 경로 문제에 대하여 기본 알고리즘에 먼저 초점을 맞출 것이다. 그리고 나서 다른 문제에 대한 다양한 확장을 검토할 것이다.

2. 원시적 Auction 알고리즘

전통적인 대칭 할당 문제에서, 1대1방식으로 연결해야 하는 n 개체와 n 대상이 있을때, 대상 j와 개체 i를 결합하면 이득은 a_{ij}가 된다. 그리고 총 이득을 최적화할 수 있도록 개체에 대상을 할당하는 것이 필요하다. 연결될 수 있는 한쌍의 집합 (i, j)을 두며, 각 개체 i에 대해서, i와 결합시킬 수 있는 집합을 $A(i)$로서 표시한다.

$A(i) = \{ j \mid (i,j) \mid \in A \}$,

그리고 각 대상 j에 대해서, j를 연결시킬 수 있는 개체의 집합을 B(j)로서 표시한다.

$B(j) = \{ i \mid (i,j) \mid \in A \}$,

할당에 의해서, 각 개체 i와 각 대상 j가, 가장 많은 하나의 쌍에 포함되는 것처럼 개체-대상의 쌍(i,j)의 집합을 S로 표시한다.

원시적 auction 알고리즘은 반복을 진행하고 가격 벡터와 할당의 결과를 초래한다. 각 반복시 처음에, 상보성 조건은 다음과 같다.

$$a_{ij} - p_{ji} = \max_{j \in A(i)} \{ a_{ij} - p_j \} \qquad (1)$$

만일 모든 개체가 할당 된다면, 알고리즘은 종료한다. 그렇지 않으면 할당 되지않은 개체 i의 비어 있지않은 부분집합 I가 선정되고, 원시적 auction 알고리즘의 전형적인 반복이 실행된다.

2-1. 원시적 auction 알고리즘의 전형적인 반복

I 를 할당되지않은 개체의 비어있지않은 부분집합에 두지.

Bidding phase(분산 계산 위상) : 각 개체 $i \in I$는 최대의 가치를 제공하는 대상을 찾는다. 즉, $j_i \in \arg \max_{j \in A(i)} \{ a_{ij} - p_j \}$ (2)

또한 분산계산 증가를 계산한다.

$$\gamma_i = v_i - w_i \qquad (3)$$

여기에서 v_i는 최고의 대상 가치이다.

$$v_i = \max_{j \in A(i)} \{ a_{ij} - p_j \} \qquad (4)$$

또한 w_i는 두번째의 대상 가치이다.

$$w_i = \max_{j \in A(i), j \neq j_i} \{ a_{ij} - p_j \} \qquad (5)$$

ji가 A(i)에서 유일한 대상이라면 v_i보다 더 작은 수나 $-\infty$가 되기 위해 w_i를 정의한다.

Assignment phase(할당 위상) : I에서 개체의 비어 있지 않은 부분 집합에 의해 최고의 대상으로서 선정된 각 대상은 가장 높은 분산 계산자를 결정한다.

$$i_j = \arg \max_{i \in p(j)} \gamma_i \qquad (6)$$

2-2. ε-상보성 auction 알고리즘

불행히도 원시 auction 알고리즘은 항상 작동하지 않으며, 분산 계산 증가 γ_i가 하나의 이상의 대상과 분산 계산자 i에 대하여 최적화를 제공할 때 영이 된다. 상보성은 상황이 끝나지 않는 싸이클을 만들면서, 여러 개체가 가격 상승없이 가장 작은 대상과 실행되는 곳에서 발생할 수 있다. [그림 1] 특히 양수 스칼라(scalar) ε를 고정 시켰고 할당과 가격 벡터 p가 상보성을 만족시킨다고 하면,

$$a_{ij_i} - p_{j_i} \geq \max_{j \in A(i)} \{ a_{ij} - p_j \} - \varepsilon \qquad (7)$$

요컨데 ε-CS를 충족시키기 위하여 모든 할당된 개체는 최고가 되는 ε이외의 대상을 할당한다. 분산 계산 증가가 적어도 항상 ε와 같도록 하며, auction 알고리즘은 분산 계산 증가 γ_i를 제외하고 원시적 auction 알고리즘과 같다.

$$\gamma_i = v_i - w_i + \varepsilon \qquad (8)$$

이러한 선택으로 ε-CS 조건은 총족된다. Auction 알고리즘에서 사용된 독특한 증가 $\gamma_i = v_i - w_i + \varepsilon$는 이러한 특성으로 최대량을

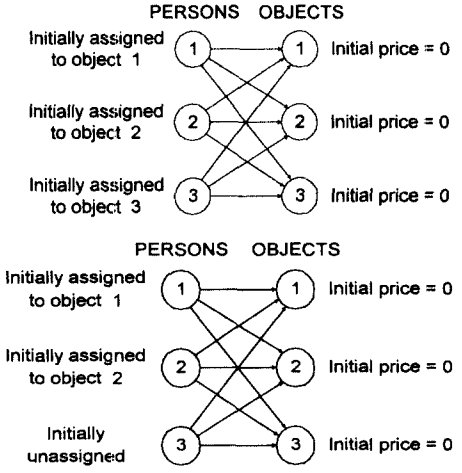
가지며, 더 작은 증가 γ_i 는 $\gamma_i \geq \varepsilon$인 동안 작동한다. 그림 2는 그림 1의 예에서 순환 문제를 이떻게 이겨 내는가를 보여준다.

At Start of Iteration #	Object Price	Assigned Pairs	Bidder	Perferred Object	Bidding Increment
1	0,0,0	(1,1), (2,2)	3	2	0
2	0,0,0	(1,1), (3,2)	2	2	0
3	0,0,0	(1,1), (2,2)	3	2	0

그림 1 : 세 개의 개체와 세 개의 대상을 위하여 결코 종결하지 않는 원시적 auction 알고리즘

할당이 최적이 되는것은 ε의 크기에 크게 의존하며, 실시간 auction에서 신중한 분산 계산자는 불필요한 높은 가격으로 대상을 얻지 않도록 과도하게 높은 분산 계산을 배열하지 않는다.

그림 3은 할당 문제의 이중 비용함수의 윤곽에 관하여 그림 1과 그림 2의 예에서 발생된 대상 가격의 결과를 보여준다. 그림에서 각 분산과 비용은 분산 계산을 받는 대상의 가격에서 주의할 때 대체로 최소화 된다. 이 결과는 보편성 있게 입증될 수 있다.[3, 6]



주어진 문제에 대해서 auction 알고리즘을 구성하기 위한 하나의 접근방식은 그것들을 할당 문제로 전환하고 auction 알고리즘을 올바르게 적용하고 계산 하는것을 합리화하는 것이다. 다음은 전통적인 최단 경로문제에 대하여 적용시킨 것이다.

각 원호(i,j)에 대해서 노드 집합 N, 원호 집합 A, 그리고 원호(i,j)를 위한 길이 a_{ij}에 그래프를 공급 한다고 가정하자. 이 부분에서 경로에 의해 (i_m, i_{m+1})이 모든 m=1,…, k-1에 대하여 원호라는 것과 같이 $(i_1, i_2, \cdots, i_k)$노드의 순서를 의미한다. 만일 추가된 노드 $i_1, i_2,\cdots,i_k$가 별개라면, 순차적$(i_1, i_2,\cdots ik)$는 단순경로(simple path)라고 부른다. 경로 길이는 그것의 원호 길이의 합계가 된다. 모든 싸이클이 양의 길이를 가진다고 가정하면서, 주어진 원점에서 시작하고(노드 1) 주어진 목적지(노드 t)에서 끝나는 모든 경로에 대하여 최단의 경로를 찾는다.

그림 4에서 보듯이 이 문제를 독특한 유형의 할당 문제로 전환 하여야하며, 등가 문제를 해결하기 위하여 auction 알고리즘을 적용할 수 있다. 그러나 이 문제의 구조는 원시적 auction 알고리즘(ε=0)의 작업이다. 모든 원호의 길이는 음수가 아니며, 영(zero) 가격 벡타와 대상 i에 대한 각 개체 i'가 for $i \neq 1$, t,에 배당하는 것과 같이 할당하는 것으로 원시

At Start of Iteration #	Object Prices	Assigned Pairs	Bidder	Preferred Object	Bidding Increment
1	0,0,0	(1,1), (2,2)	3	2	ε
2	0, ε,0	(1,1), (3,2)	2	1	2 ε
3	2 ε, ε,0	(2,3), (3,1)	1	2	2 ε
4	2 ε,3 ε,0	(1,2), (2,1)	3	1	2 ε
5	4 ε,3 ε,0	(1,3), (3,2)	2	2	2 ε
6	. . . .	. . . . .	. . . .	. . . .	. . . .

그림 2 : 그림 1의 예에서 싸이클 문제를 해결한 auction 알고리즘

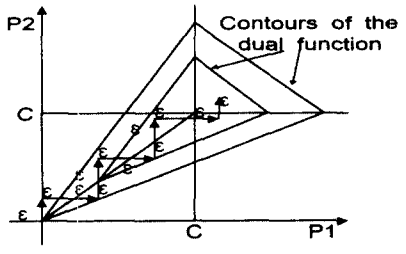


그림 3 : 그림 1의 예에서 auction 알고리즘에 의한 가격 p_1과 p_2의 순차적 생성 과정

3. 최단 경로 문제에 대한 auction 알고리즘의 응용

3-1 최단 경로 문제

적 auction 알고리즘을 시작할 수 있다.

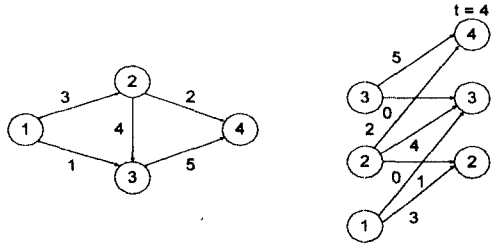


그림 4 : 최단 경로 문제(원점 = 1, 도착점 t= 4)와 최단 경로 문제의 대응 할당 문제

여기에서 결과는 경로 p=(1, i_1, ..., i_k)와 일치한다. $i_k \neq t$, 로 인한 원시 auction 알고리즘에서 할당이 되지 않은 개체 경로는 종단 노드에 대응하는 개체 i'_k이다. 할

당할 수 없는 개체 i'_k는 분산 계산을 제안하고 두 개의 가능성을 가진다.

(a) i'_{k-1}은 할당할 수 없게 되고 새로운 할당에 상응하는 경로$(1,i_1, \cdots, i_{k-1})$은 하나의 노드에 의해 "축소"를 거쳐서 이전의 경로 $(i, I_1, I_2, \cdots, i_k)$로부터 얻을 수 있는 경우에 최고의 대상은 i_k이다.

(b) 최고의 대상은 $i_{k+1} \neq i_k$이다.

3-2. Auction의 전형적인 반복/최단 경로 알고리즘

CS를 충족하는 다른 쌍에서, CS를 충족 시키는 하나의 쌍(P,p)으로 변형하면서 알고리즘은 계속 반복한다. 각 반복에서, 경로 P는 새로운 노드에 의해 확장하거나 종단 노드를 삭제함으로써 축소된다. 그러므로 종단 노드의 가격은 증가하며, 이러한 경우에 경로는 확장되거나 가격 P로 변화되지 않는다. 반복은 다음과 같다. 즉 i를 P의 종단 노드에 두자.

If $p_i < \min_{(i,j)\in A} \{a_{ij} + p_j\}$,

go to step 1; else step 2.

step 1 (수축 경로) :

Set $p_i := \min_{(i,j)\in A} \{a_{ij} + p_j\}$,

and if $i \neq 1$, contract P. Go to the next iteration.

step 2 (확장 경로) : Extened P by node j_i where

$$p_i = \arg \min_{(i,j)\in A} \{a_{ij} + p_j\},$$

If j_i is the destination t, stop; P is the desired shortest path. otherwise, go to the next iteration.

P는 1에서 j_i 까지 단순경로이다; 만약 그렇지 않으면 j_i에 P를 더하는 것은 싸이클을 생성하는 것이다, 그리고 이러한 싸이클의 모든 원호(i,j)에 대하여 $p_i = a_{ij} + p_j$ 가진다. 상보성 조건은 $p_i - p_j \leq a_{ij}$로 그림 5(b)에서 보여준 모든 원호(i,j)에 유지된다. 만일 모델이 선택되고, 원점 노드로 부터 매달린채라면 $p_i - p_j = a_{ij}$ 가지고있다. 따라서 그림 5(c)에서 보인 것처럼 최단 경로와 종단 노드 사이에 연결된 문자열의 대응은 빈틈없이 연결된다. 특히 다른 노드 i에 대하여 원점 노드 1과 관련된 빈틈없는 연결 길이는 p_1-p_i이고, 또한 1에서 i까지 최단 거리와 동일하다. 이러한 결과는 최소 경로/최대 팽창력 정리이다.[6, 7]

각 노드는 하나의 볼이다. 그리고 모든 원호 $(i,j)\in A$, 노드 i와 j는 a_{ij} 길이의 문자열과 연결되고,

$p_i - p_j \leq a_{ij}$를 충족한 노드의 p_i 수직 좌표이고, (a)에서 주어진 문제를 위하여 그림(b)에서 보여준다. 그 모델은 원점 노드 1로부터 좌측에 매달리고 선택되며, $p_1 - p_i$은 그림(c)에서 보여 주듯이 각 노드 i의 최단 거리이다.

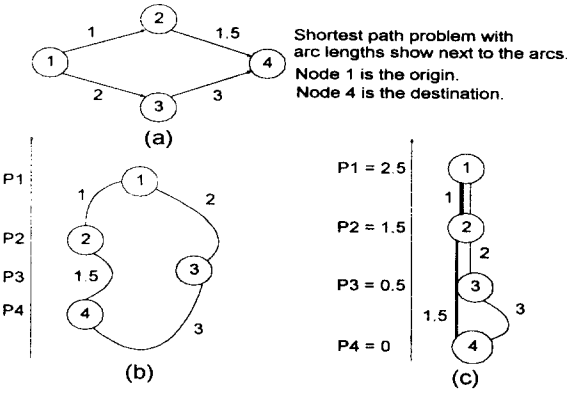


그림 5 : 최단 경로 문제를 위한 CS조건

Auction/최단경로 알고리즘은 ball-and-strings 모델의 관점에서 해석될 수 있다.[9] 그것은 노드가 그림 6에서 어떻게 각 단계를 거슬러 올라 갔는지를 과정으로 보여줄 수 있다. 처음에 모든 노드는 판판한 표면에 정지하고 있다. 각 단계에서 적어도 하나 이상의 문자열이 이웃 되는 수준으로 원점에서 시작하여 빈틈 없는 연결에서 마지막 노드로 상승한다.

단일 원점과 복합 목적지가 같을 때 알고리즘은 사실상 변화없이 적용될 수 있다. 모든 목적지가 적어도 한 번 경로 P의 종단 노드가 될 때 알고리즘을 종료한다. 원점과 목적지의 역할이 각 원호의 방향을 먼저 교환 함으로써, 알고리즘은 복합 원점과 단일 목적지의 문제를 적용할 수 있다.

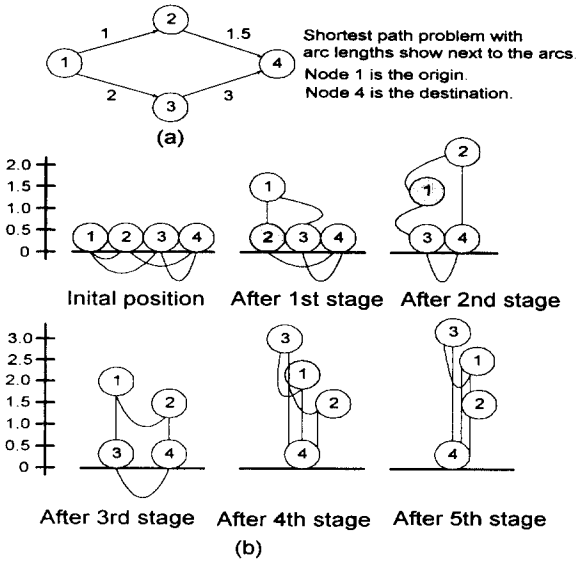


그림 6 : (a)에서 보여준 문제에 관해서 ball-and-strings 모델의 관점에서 auction/최단 경로 알고리즘

3-3. 실제 계산적 측면에서의 최단 경로문제

개선된 auction/최단 경로 알고리즘과 역 계수부의 결합을 제외하고,[10] 다수의 유용한 구현 아이디어중 알고리즘의 계산적 병목 현상은 다음과 같은 식으로 계산한다.

$$\min_{(i,j)\in A}\{a_{ij}+p_j\} \qquad (11)$$

그것은 노드 i가 경로의 종단 노드가 삽입될 때마다 종료한다. CS 조건 $p_i \le a_{ij}+p_j$가 모든 원호(i,j)에 대하여 항상 유지되기 때문이다. (i,j_i)는 (식: 12)을 충족시키며, 그 다음으로 (식; 13)도 만족시킨다.

$$p_i = a_{ij_i}+p_{j_i} \qquad (12)$$

$$a_{ij_i}+p_{j_i} = \min_{(i,j)\in A}\{a_{ij}+p_j\} \qquad (13)$$

만약 i가 경로의 종단 노드가 된다면 경로는 j_i에 의해 확장될 수 있다. 이것은 다음과 같은 구현 전략을 제안한다: 경로가 종단 노드가 되는 i에서 발생할 때마다 (식; 14)와 같이 계산되며, 원호(i,j_i)와 함께 (식; 15)를 구한다.

$$\min_{(i,j)\in A}\{a_{ij}+p_j\} \qquad (14)$$

$$j_i = \arg\min_{(i,j)\in A}\{a_{ij}+p_j\}. \qquad (15)$$

다음으로, 노드 i는 경로의 종단 노드가 된다. 조건 $p_i=a_{ij}+p_{ji}$가 충족되는지를 검토하며, 만약 그렇다면, $\min_{(i,j)\in A}\{aS_{ij}+p_j\}$의 계산을 통해서 노드 ji에 의한 경로를 확장한다.

"가장 가까운 이웃"를 계산 한다고 가정해보자.

$$j_i = \arg\min_{(i,j)\in A}\{a_{ij}+p_j\} \qquad (16)$$

"두번째 가장 가까운 이웃" k_i를 가지고 계산해보자

$$k_i = \arg\min_{(i,j)\in A, j\neq j_i}\{a_{ij}+p_j\} \qquad (17)$$

그리고 대응되는 "두번째 최고 수준"을 가정해보자

$$\omega_i = a_{ik_i}+p_{k_i} \qquad (18)$$

따라서 다음으로 노드 i는 경로의 종단 노드가 된다. 조건 $a_{ij_i}+p_{j_i} \le \omega_i$가 충족되는지를 검토할 수 있다. 그리고 충족되었다면, j_i가 여전히 공식에서 최소화를 달성한다는 것을 알 수 있다.

$$\min_{(i,j)\in A}\{a_{ij}+p_j\} \qquad (19)$$

이 최소화의 계산을 없애면서, 한편으로, $a_{ij_i}+p_{j_i} > \omega_i$를 가진다. ($\omega_i$ 계산 다음으로 P_j의 증가에 기인하여), $\omega_i = a_{ik_i}+p_{k_i}$를 점검할 수 있다. 만일 그렇다면, k_i는 "가장 가까운 이웃"이 된다.

$$k_i = \arg\min_{(i,j)\in A}\{a_{ij}+p_j\}, \qquad (20)$$

올바른 구현으로, 위에서 언급한 장치는 3에서 5까지의 범위내에 있는 요소에 의해 식 $\min_{(i,j)\in A}\{a_{ij}+p_j\}$의 수를 감소할 수 있다. 위의 장치는 단일 원점/임의적으로 생성되는 문제의 모든 도착점 때문에 가장 빠른 auction/최단 경로 코드에서 사용된다.[8]

4. 결 론

선형 망 흐름 문제의 변화에 관하여 많은 연구가 진행되어 왔다. Auction 알고리즘을 확장하고 적용하기 위하여 전통적인 방법으로 첫째 최단 경로와 최단 경로 개선 방법과 둘째 비용과 비용 개선방법을 동일점(equal footing)에 배치하기 위하여, auction 알고리즘은 아직은 완전히 개발된 것은 아니며, 더나은 연구를 위하여, 전통적인 방법으로 골고루 적용하여 더나은 경쟁력을 갖추오록 하여야한다.

이 논문에서 논의된 auction 알고리즘은 컴퓨터 코드를 구현해 왔다. ε-상보성 auction 알고리즘은 각 반복 초기에 ε-CS를 충족하는 할당과 이득 벡타 π를 포함하고 있으며, 분산 계산 위상과 할당 위상을 결정한다. 또한 선형 망 흐름 문제에서 auction 알고리즘을 이용하여 ε-CS조건을 충족하는 전형적인 반복과 최단 경로를 구현하고, 가장 가까운 이웃 노드의 경로를 최소화 하는 방법으로 실제적 계산적 측면에서 최단 경로를 구현하는 방법을 연구하고자 하였다.

앞으로의 연구과제는 k 최단 경로 문제와 k 노드-병행 최단 경로 문제를 이용하여 선형망 구조를 더욱 간략화 하며, 최소 비용 흐름 문제를 이용한 일반적 auction 알고리즘의 적용에 대한 것이다.

참고 문헌

[1] : Bertsekas D. P., and Mitter, S.K., "Descent Numerical Methods for Optimization Problems with Nondifferentiable Cost Functions," SIAM Journal on Control, Vol. 11, pp. 637-652.

[2] : Bertsekas, D. P., " A Distributed Asynchronous Relaxation Algorithm for the Assignment Problem." Proc. 24th IEEE Conf. Dec. & Contr., 1985, pp. 1703-1704.

[3] : Bertsekas, D. P., "The Auction Algorithm: A Distributed Relaxation Method for the Assignment Problem, " Annals of Operations Research, Vol. 14, 1988, pp. 105-123.

[4] : Bertsekas, D. P., and Eckstein, J., "Dual Coordinate Step Methods for Linear Network Flow Problems," Math. Progr., Series B, Vol. 42, 1988, pp. 203-243.

[5] : Bertsekas, D. P., and Tsitsiklis, J. N., Parallel and

Distributed Computation : Numerical Methods, Prentice-Hall, Englewood Cliffs, N. J., 1989.

[6] : Bertsekas, D.P., Linear Network Optimization: Algorithms and Codes, M.I.T. Press, Cambridge, Mass., 1991.

[7] : Rockafellar, R T., Network Flows and Monotropic Programming, Wiley-Interscience, N. Y., 1984.

[8] : Bertsekas, D. P., Pallottino, S., and Scutella', M. G., "Polynomial Auction Algorithms for Shortest Paths." submitted for publication.

[9] : Minty, G. J., "Monotone Networks," proc. Roy. Soc, London, A, Vol. 257, 1960, pp. 194-212.

[10] : Dimitri P. Bertsekas, "Auction Algorithm for Network Flow Problems", Laboratory for Information and Decision System, M.I.T, Cambridge, Mass. May 1992, LIDS-P-2108.