Génie Industriel

THEORETICAL EFFICIENCY OF THE ALGORITHM "CAPACITY" FOR THE MAXIMUM FLOW PROBLEM.

Maurice QUEYRANNE
Département de Génie Industriel
Ecole Polytechnique de Montréal.

RAPPORT TECHNIQUE NO. EP 78-R-43

September 1978.

Ecole Polytechnique de Montréal

Campus de l'Université
de Montréal
Case postale 6079
Succursale 'A'
Montréal, Québec
H3C 3A7

THEORETICAL EFFICIENCY OF THE ALGORITHM "CAPACITY" FOR THE MAXIMUM FLOW PROBLEM.

DON

Maurice QUEYRANNE

Département de Génie Industriel

Ecole Polytechnique de Montréal.

RAPPORT TECHNIQUE NO. EP 78-R-43

September 1978.

THEORETICAL EFFICIENCY OF THE ALGORITHM "CAPACITY" FOR THE MAXIMUM FLOW PROBLEM*

Maurice QUEYRANNE

Ecole Polytechnique de Montréal.

ABSTRACT: The algorithm Capacity defined by Edmonds and Karp, is an augmenting path algorithm which yields the maximum increase in flow value at each iteration. For networks with real capacities, we show that Capacity may require an infinite number of iterations, and an example of a "bad" network is produced. This result contrasts with the existence of other less "greedy" augmenting path algorithms which are always finite. However, we show that the sequence of flows constructed by Capacity converges toward a maximum flow. For networks with integer capacities, with a arcs and average capacity $\bar{c}$, Edmonds and Karp's upper bound of the order of $O\ (a(\log a + \log \bar{c}))$ is derived and a class of networks is produced for which Capacity requires this number of iterations.

AMS 1970 subject classification. Primery: 90C35.

IAOR 1973 subject classification. Main: Network Programming.

Key words. Network flows, Maximum flow problem, Algorithms, Computational Complexity, Worst-case analysis.

* This research was partially supported by the National Research Council of Canada, Grant A-4592.

On the algorithm "Capacity" for the maximum flow problem.

1- INTRODUCTION

Consider a finite directed network $N=(V,A,c)$, where V is the set of vertices, A is the set of arcs (we allow multiple parallel arcs) and c is a positive real-valued function defined on A. For every arc a, we define its head (or destination) $h(a)$, its tail (or origin) $t(a)$ and its capacity $c(a)$. Given two vertices s and t called the source and the sink, we call an (s,t) - flow (or more simply a flow) any real-valued function x defined on A satisfying the (Kirchhoff) conservation law

$$\sum_{a:h(a)=i} x(a) = \sum_{a:t(a)=i} x(a)$$

for all vertices $i \in N$ distinct from s and t. By summing all these equations, we have

$$\sum_{a:h(a)=t} x(a) = \sum_{a:t(a)=s} x(a) = v(x)$$

and we call this quantity $v(x)$ the value of the flow x. If in addition $0 \leqslant x(a) \leqslant c(a)$ for all arcs a, we call x a feasible flow.

The maximum flow problem is the problem of finding a feasible flow of maximum value. Among the numerous algorithms which have been proposed for solving this problem (see [6] for a recent review), we focus on those based on augmenting paths. Given a feasible flow x we call augmenting network $N_x=(V,A_x,c_x)$ a network with the same vertex set V as N, and with arc set A_x deduced from A by

$$(a \in A \text{ and } x(a) < c(x)) \Rightarrow (a \in A_x \text{ and } c_x(a) = c(a) - x(a)) \quad (1)$$

$$\text{and} \quad (a \in A \text{ and } x(a) > 0) \Rightarrow (\bar{a} \in A_x, h_x(\bar{a}) = t(a),\ t_x(\bar{a}) = h(a) \text{ and } c_x(\bar{a}) = x(a)) \quad (2)$$

Note that $|A_x| \leqslant 2\,|A|$

We call direct arcs the arcs defined by (1) and reverse arcs the arcs defined by (2). An augmenting path P relative to a feasible flow x, is an elementary (s,t) - path in N_x, and its capacity $c_x(P)$ is the smallest of the values $c_x(a)$ for all the arcs, direct or reverse, in this path. If a feasible flow x admits an augmenting path P, x is not a maximum flow since we can define a flow x' with value $v(x') = v(x) + c_x(P)$ by increasing (resp. decreasing) the flow on the direct (resp. reverse) arcs of P by the amount $c_x(P)$. On the other hand if x does not admit an augmenting path, it follows from a theorem by Ford and Fulkerson [3] that x is a maximum flow. An augmenting path algorithm starts with any feasible flow x (for instance x=0), seeks an augmenting path, then modifies x, and iterates this process until a flow is attained such that no augmenting path exists. When all the capacities and the initial flow are integer, it is known that any augmenting path algorithm finds a maximum flow within at most $v(x^*)$ iterations, where x^* is any maximum flow. When real capacities are allowed, it is possible that an augmenting path algorithm may not terminate in a finite number of steps, (see [3], [7] for two examples). For these two examples, the sequence of flows constructed by the algorithm converges toward a flow which is still not maximum. On the other hand, Tucker has shown [8] that any "consistent" augmenting path algorithm must produce a maximum flow in a finite number of iterations.

The algorithm Capacity defined by Edmonds and Karp [2] is an augmenting path algorithm which uses at every iteration an augmenting path with largest capacity, producing the maximum possible increase in the flow value per iteration. When the capacities and the initial flow are integer they show that the number of iterations is bounded by $O(|V|^2(\log|V| + \log \bar{c}))$ where $|V|$ is the number of vertices and $\bar{c}$ denotes the average capacity of an arc in the network. Since an augmenting path with maximum capacity can be found within polynomial time, it follows that Capacity is a "good algorithm" in the sense of Karp [4]. However, Zadeh [9] states that "at the present, no bound independent of the capacities is known for Capacity". It is the purpose of this paper to show that no such bound can exist and that Edmonds and Karp's bound is tight.

In the first section we produce a network with real capacities on which Capacity cannot find a maximum flow within a finite number of iterations. In section 2 we show that, in contrast with previously known non-finite augmenting path algorithms [3], [7], the sequence of flows constructed by Capacity converges toward a maximum flow. As a corollary we obtain for networks with integer capacities an $O(|A|(\log|A| + \log \bar{c}))$ bound on the number of iterations, which is similar to the bound of Edmonds and Karp. In section 4, we construct a class of networks with integer capacities for which this bound is tight.

2. A bad network for Capacity

In this section we will denote an arc a by (t(a), h(a)), the choice of the precise arc among the multiple arcs having same endpoints being clear from the context.

Consider the network N=(V,A,c) given in Figure 1. It has eight nodes, namely the source s, the sink t and six intermediate nodes numbered from 1 to 6, and nineteen arcs, classified as follows:

special arcs:	(1,2) and (3,4) with capacity r
	(5,6) with capacity 1
supplementary arcs:	(s,5) and (6,t) with capacity 1
	(s,1), (2,4), (3,6) and (5,t) with capacity $S_1 = \frac{1}{2} r + \frac{1}{2}$
	(s,3) and (4,t) with capacity r
	(s,2), (1,3), (4,6) and (5,t) with capacity $S_2 = \frac{1}{2}$
	(s,2), (1,4), (3,5) and (6,t) with capacity $S_3 = \frac{1}{2} r$

where $r = \frac{1}{2} (\sqrt{5} - 1) = .61803398....$ We note that this network includes, for ease of demonstration, multiple arcs. These multiple arcs may be eliminated by including some additional nodes.

Before describing the application of Capacity to this network, we recall some useful properties related to the number r:

$$r^{i-1} - r^{i} = r^{i+1} \quad \text{all } i \geqslant 1$$

$$S_3 = \sum_{k=0}^{\infty} r^{3k+3} = \frac{1}{2} r = .3090169...$$

$$S_2 = \sum_{k=0}^{\infty} r^{3k+2} = \frac{1}{2} = .5$$

$$S_1 = \sum_{k=0}^{\infty} r^{3k+1} = \frac{1}{2} r + \frac{1}{2} = .8090169...$$

$$S_1 \geq r \geq S_2 \geq r^2 \geq S_3 \geq r^3 \geq S_1 - r \geq r^4 \geq S_2 - r^2$$

The first path identified by Capacity is (s,5),(5,6),(6,t) with capacity 1, and it is unique. After sending this unit of flow along this path, we discover three paths with capacity r:

(s,3), (3,4), (4,t)

(s,1), (1,2), (2,4), (4,t)

and (s,3), (3,6), $\overline{(5,6)}$, (5,t) (recall that $\overline{(i,j)}$ denotes the reverse arc (j,i) associated with (i,j) in the augmenting network). For the present, we assume that we select (s,3), (3,4), (4,t). At this point the arcs (s,3), (s,5), (4,t) and (6,t) are saturated and their flow will never be modified.

At the third iteration a cyclic pattern starts. The value of flows in the various arcs at iterations 3k, 3k+1, 3k+2 (where k is an arbitrary positive integer) are displayed in Table 1.

beginning of iteration	Special arcs			Supplementary arcs with capacity		
	(1,2)	(3,4)	(5,6)	S_1	S_2	S_3
3k	0	r^{3k-2}	r^{3k-3}	$S_1(1-r^{3k-3})$	$S_2(1-r^{3k-3})$	$S_3(1-r^{3k-3})$
3k + 1	r^{3k-2}	0	r^{3k-1}	$S_1(1-r^{3k})$	$S_2(1-r^{3k-3})$	$S_3(1-r^{3k-3})$
3k + 2	r^{3k}	r^{3k-1}	0	$S_1(1-r^{3k})$	$S_2(1-r^{3k})$	$S_3(1-r^{3k-3})$

Table 1: Flows in the arcs during the cycle.

At the iteration 3k, the unique augmenting path with maximum capacity is (s,1), (1,2), (2,4), $\overline{(3,4)}$, (3,6), $\overline{(5,6)}$, (5,t), with capacity r^{3k-2}. This path contains the special arcs and the arcs with (initial) capacity S_1. The flow on these arcs becomes $S_1(1-r^{3k-3}) + r^{3k-2} = \sum_{i=k}^{\infty} r^{3i+1} = S_1(1-r^{3k})$.

At the iteration 3k+1, the unique path identified by Capacity is (s,2), $\overline{(1,2)}$, (1,3), (3,4), (4,6), $\overline{(5,6)}$, (5,t) with capacity r^{3k-1}, using the special arcs and the arcs with capacity S_2. The flow on these arcs becomes $S_2(1-r^{3k})$. At the iteration (3k+2), the unique path identified by Capacity is (s,2), $\overline{(1,2)}$, (1,4), $\overline{(3,4)}$, (3,5), (5,6), (6,t) with capacity r^{3k}, using the special arcs and the arcs with capacity S_3. The flow on these arcs becomes $S_3(1-r^{3k})$. At this point, i.e. iteration 3k + 3, a pattern similar to the one assumed at iteration 3k is obtained. Hence, the algorithm Capacity cycles, producing on increase of r^{i-2} in the flow value ~~at the i-th~~ iteration (i $\geqslant$ 2).

At the second iteration, there are two other possible choices. Note that all the tie-breaking algorithms that could be defined using Zadeh's terminology [9], namely Capacity / Short / Reverse and Capacity / Reverse / Short would make the same choice as before. If a version of Capacity selects (s,1), (1,2), (2,4), (4,t) at the second iteration, the next path would be (s,3), (3,6), $\overline{(5,6)}$, (5,t) and is unique. On the other hand, if the second path selected is (s,3), (3,6), $\overline{(5,6)}$, (5,t) the third path must be (s,1), (1,2), (2,4), (4,t). In both cases, the flow which results is the same as the one obtained after the third iteration in the previous development, and then the same cyclic pattern occurs.

Finally note that, since the network is planar, Berge's algorithm [1] applies and produces the maximum flow in nine augmentations.

3- Convergence:

In the previous example, the sequence of flows produced by Capacity converges to the (unique) maximum flow. By reference to the behavior of other non-finite algorithms for maximum flow [3], [7], which may converge to a solution which is not a maximum flow, we prove that Capacity is a valid algorithm, at least in this respect.

Theorem 1: The algorithm Capacity produces a (finite or infinite) sequence of flows which converges to a maximum flow.

Proof: to prove this theorem, we use the following well-known Lemma.

Lemma 2: Given a network and a feasible flow, there exist a sequence of at most $|A|$ augmenting paths which yields a maximum flow.

Sketch of proof: One proof of this lemma uses the arc-path incidence matrix of the augmenting network associated with the given feasible flow, and parallels the proof of Theorem 4.3, Chapter 4 in [5]. Another possible proof is constructive: let x be a feasible flow and x^* be a maximum flow (without circuit), and consider the flow $x^* - x$ as a feasible flow in the augmenting network associated with x: identify an elementary (s-t) path with positive flow $x^* - x$ and substract their minimum from the flows on the arcs of this path, eliminating at least one arc, and so on until the flow is exhausted. Note that if the initial flow has some circuits, we may consider a maximum flow x^* with the same amount of flow on these circuits. #

We need prove Theorem 1 only for an infinite sequence of flows. Let $v^1, v^2, \ldots, v^k, \ldots$ denotes the sequence of values of these flows. Since this sequence is increasing and bounded above by the capacity of any cut, it converges. Assume that its limit v is not the maximum value v^* of a flow. Since all the flows constructed are feasible, then $v^* - v > 0$. Thus there is an integer h such that

$$v^{h+1} - v^h < \frac{v^* - v}{|A|}$$

Now by applying the above lemma to the flow x^h obtained at this hth interation, we know that there exists an augmenting path with capacity greater than or equal to $\frac{v^* - v^h}{|A|}$, hence the value of the next flow will be

$$v^{h+1} \geqslant v^h + \frac{v^* - v^h}{|A|} \geqslant v^h + \frac{v^* - v}{|A|}$$

a contradiction. #

This results extends to networks with some infinite capacities: if there is a cut with finite capacity separating s from t, then the previous theorem applies; otherwise, Capacity finds an infinite flow in exactly one iteration.

4. Worst-case behavior for networks with integer capacities.

If the capacities (and the initial flow) are integer, we already know (e.g. [3]) that the number of iterations of Capacity will be finite. We can derive an upper bound on this number, similar to the one obtained by Edmonds and Karp, from the developments in the previous section.

Corollary 3 (Edmonds and Karp [2])

If the capacities and the initial flow are integer, the algorithm Capacity produces a maximum flow in at most $O(|A| \max(\log |A|, \log \bar{c}))$ iterations, where $\bar{c}$ denotes the average capacity of an arc in the network.

Proof: in the proof of the previous theorem, we have shown that:

$$v^{h+1} \geqslant v^h + \frac{v^*-v^h}{|A|}$$

hence $$v^*-v^{h+1} \leqslant (v^* - v^h)\left(1 - \frac{1}{|A|}\right)$$

Following the proof of Edmonds and Karp, we obtain by induction

$$v^* - v^h \leqslant v^* \left(1 - \frac{1}{|A|}\right)^h$$

If the capacities and flow are integer, and if v^h is not a maximum flow then:

$$v^* - v^h \geqslant 1 \quad \text{so} \quad h \leqslant \frac{\log v^*}{-\log\left(1 - \frac{1}{|A|}\right)} .$$

Since $v^* \leqslant |A|\,\bar{c}$ and $-\log\left(1 - \frac{1}{|A|}\right) \geqslant \frac{1}{|A|} - \frac{1}{2|A|^2}$ for $|A| \geqslant 1$

it follows that $h \leqslant \dfrac{\log |A| + \log \bar{c}}{\dfrac{1}{|A|} - \dfrac{1}{2|A|^2}}$

which is an $O(|A|(\log |A| + \log \bar{c})$ bound. #

In the case of networks with some infinite capacities, the term $\log |A| + \log \bar{c}$ may be replaced by the logarithm of the finite capacity of any cut, if it exists. The bound obtained by Edmonds and Karp is similar, with the term $|A|$ being replaced by M, an upper bound on the number of arcs in an indirected cut separating s and t. However, this number M is usually harder to compute than to solve the maximum flow problem and furthermore, these bounds are equivalent in a worst-case sense.

In order to provide a class of networks with integer capacities to which the application of Capacity leads to $0(|A|(\log |A| + \log \bar{c}))$ steps, we begin by displaying a class of networks with fixed number of nodes and arcs, requiring $0(\log \bar{c})$ iterations of Capacity. For an integer $q \geqslant 1$ let $N^q = (V,A,c^q)$ where V and A are the vertex and arc sets of the "bad" network discussed in section 1, and $c^q(a) = \lfloor 2^q c(a) \rfloor$ where $\lfloor x \rfloor$ denotes the largest integer less than or equal to x, and $c(a)$ is the (real) capacity of the arc a in this bad network. This is a class of networks with integer capacities, with $\log \bar{c} = 0(q)$.

<u>Proposition 4</u>: The application of the algorithm Capacity to the maximum flow problem in a network N^q requires a number of iterations of the order of $0(q)$ to find a maximum flow.
The proof of this proposition is given in Appendix.

We use this class N^q of networks to construct a class of networks

$N^{vq} = (V^{vq}, A^{vq}, c^{vq})$ where v is an integer ≥ 1.

These networks are defined as follows:

$$V^{vq} = \left\{(i,j) : i \in V,\ j \in \{1,\dots,v\}\right\} \cup \{\bar{s},\bar{t}\}$$

where $\bar{s}$ and $\bar{t}$ are respectively the source and sink of N^{vq}

$$A^{vq} = \Big\{(a,(i,j)) : a \in A,\ (i,j) \in \{1,\dots,v\}^2,\ h(a,(i,j)) = (h(a),j) \text{ and } t(a,(i,j)) = (t(a),i)\Big\}$$
$$\cup \left\{(\bar{s},i) : i \in \{1,\dots,v\},\ h(\bar{s},i) = (s,i),\ t(\bar{s},i) = \bar{s}\right\}$$
$$\cup \left\{(j,\bar{t}) : j \in \{1,\dots,v\},\ h(j,\bar{t}) = \bar{t},\ t(j,\bar{t}) = (t,j)\right\}$$

and $c^{vq}(a,(i,j)) = c^q(a) = \lfloor 2^q c(a) \rfloor$ for all $a \in A$, $(i,j) \in \{1,\dots,v\}^2$

$$c^{vq}(\bar{s},i) = c^q(j,\bar{t}) = v2^{q+2}$$

Verbally, these networks are deduced from N^q by reproducing each vertex v times, by replacing each arc by the complete bipartite graph $K_{v,v}$ and assigning to each such arc the capacity of the corresponding arc in N^q. To this network are added a "supersource" $\bar{s}$ and a "supersink" $\bar{t}$, respectively connected to the "old" sources and sinks by arcs with sufficient capacities. We have $|V^{vq}| = 8v+2$,

$$|A^{vq}| = 19v^2 + 2v \text{ and } O(\log \bar{c}^{vq}) \doteq O(\log \bar{c}^q) = O(q)$$

To every (s,t) - chain in N^q we may associate v^2 arc-disjoint $(\bar{s},\bar{t})$-chains in N^{vq}. Hence the application of the algorithm Capacity to N^q can be described as sequences of v^2 consecutive augmenting paths corresponding to the augmenting paths identified by Capacity in N^q, provided that the augmenting paths selected at the iterations $v+1, v+2, \dots, 2v$ correspond to the same second augmenting path in N^q. In that case, we obtain:

Proposition 5: The application of Capacity to N^{vq} requires about $v^2 O(q)$ iterations, that is $O(|A| \log \bar{c})$ iterations.

Finally, the presence of the term $\log |A|$ in the bound on the number of iterations deserves some comments. Since any augmenting path algorithm requires at most $O(|A|\bar{c})$ iterations to find a maximum flow in a network with integer capacities, it appears that this term $\log|A|$ plays a role only for classes of networks such that

$$O(\log |A|) \lessdot O(\bar{c}) < O(|A|).$$

For classes of networks such that $\bar{c}$ increases more slowly than $\log |A|$, the best known bound on the number of iterations is still $O(|A|\bar{c})$ for Capacity, the same as for any augmenting path algorithm.

REFERENCES

[1] C. Berge et A. Ghouila-Houri (1962). Programmes, jeux et réseaux de transport. Dunod, Paris. (English translation (1976). Programming,Games and Transportation Networks, Methuen, London).

[2] J. Edmonds and R.M. Karp (1972). Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems. J. Assoc. Comput. Mach. 19, 248-264.

[3] L.R. Ford and D.R. Fulkerson (1962). Flows in Networks. Princeton Univ. Press, Princeton, N.J.

[4] R.M. Karp (1972). Reducibility among Combinatorial Problems. Complexity of Computer Computations (R.E. Miller and J.W. Thatcher, eds.). Plenum Press, New York, N.Y.

[5] E.L. Lawler (1976). Combinatorial Optimization: Networks and Matroids. Holt, Rinehart and Winston, New York, N.Y.

[6] E.L. Lawler (1977). Shortest Paths and Network Flow Algorithms. Presented at Discrete Optimization 77, Vancouver, B.C.

[7] J. Ponstein (1972). On the Maximal Flow Problem with Real Arc Capacities. Mathematical Programming 3, 254-256.

[8] A. Tucker (1977). A Note on Convergence of the Ford-Fulkerson Flow Algorithm. Mathematics of Operations Research 2, 143-144.

[9] N. Zadeh (1973). More Pathological Examples for Network Flow Problems. Mathematical Programming 5, 217-224.

Appendix: Proof of Proposition 4

It is easily verified that, for all $q \geqslant 4$, the algorithm Capacity uses, at the fourth iteration, the path

$$(s,2),\ (\overline{1,2}),\ (1,3),\ (3,4),\ (4,6),\ (\overline{5,6}),\ (5,t)$$

which belongs to the cycle.

Hence assume that $q \geqslant 4$ and denote by k the number of iterations which are identical when Capacity is applied to N^q and to the bad network N. Since $k \geqslant 4$, this k-th iteration is in the cycle. Denote by x_{hq} (resp x_h) the flow in N^q (resp N) after the h-th augmentation (that is at the beginning of the (h + 1)st iteration).

For commodity of demonstration, we number the special arcs as follows:

$$a_1 = (1,2),\ a_2 = (3,4),\ a_3 = (5,6).$$

Lemma 6 : Let h be an integer such that $4 \leqslant h \leqslant k$, and i, j, ℓ be integers in $\{1,2,3\}$ such that $i \equiv h \pmod 3$, $j \equiv h+1 \pmod 3$ and $\ell \equiv h+2 \pmod 3$.

If h is odd, then:

$$2^q r^{h-1} \leqslant x_{hq}(a_i) \leqslant 2^q r^{h-1} + F_{h-1}$$

$$2^q r^{h-2} - F_{h-2} \leqslant x_{hq}(a_j) \leqslant 2^q r^{h-2}$$

$$x_{hq}(a_\ell) = 0.$$

If h is even, then:

$$2^q r^{h-1} - F_{h-1} \leqslant x_{hq}(a_i) \leqslant 2^q r^{h-1}$$

$$2^q r^{h-2} \leqslant x_{hq}(a_j) \leqslant 2^q r^{h-2} + F_{h-2}$$

$$x_{hq}(a_\ell) = 0,$$

where F_i denotes the i-th Fibonacci number, defined by

$$F_o = 0,\ F_1 = 1,\ F_i = F_{i-1} + F_{i-2} \text{ for } i \geqslant 2.$$

Proof: Since $2^q r - 1 \leqslant \lfloor 2^q r \rfloor \leqslant 2^q r$

and $2^q r^2 \leqslant 2^q - \lfloor 2^q r \rfloor \leqslant 2^q r^2 + 1$

the lemma is verified for $h = 4$.

Whenever $h \leqslant k$, Capacity follows the same paths in N^q as in N.

Assume the lemma holds for $h \geqslant 4$ and consider the augmenting path P_{h+1} generated at the iteration $h+1 \leqslant k$. This augmenting path includes the special arcs a_i and a_ℓ as direct arcs, and a_j as reverse arc, and the supplementary arcs with initial capacity S_ℓ as direct arcs. The flow driven along this path in N^q is the flow on a_i (since $h + 1 \leqslant k$, as in N) and the resulting flows verify, when h is odd:

$$x_{h+1,q}(a_i) = 0$$

$$2^q r^{h-2} - F_{h-2} - 2^q r^{h-1} - F_{h-1} \leqslant x_{h+1,q}(a_j) \leqslant 2^q r^{h-2} - 2^q r^{h-1}$$

so $2^q r^h - F_h \leqslant x_{h+1,q}(a_j) \leqslant 2^q r^h$

and $2^q r^{h-1} \leqslant x_{h+1,q}(a_\ell) \leqslant 2^q r^{h-1} + F_{h-1}$

After properly redefining i, j and ℓ, the lemma is verified since h+1 is even.

The argument is similar for h even and thus the lemma is proven #

Lemma 7: Let h be an integer such that $4 \leqslant h \leqslant k$. For any supplementary arc a, its "remaining capacity" $c_q(a) - x_{hq}(a)$ in N^q after the k-th augmentation is bounded by

$$2^q(c(a)-x_h(a))-F_{h-2}-1 \leqslant c_q(a)-x_{hq}(a) \leqslant 2^q(c(a)-x_h(a)) + F_{h-1}$$

when h is odd, and by

$$2^q(c(a)-x_h(a))-F_{h-1}-1 \leqslant c_q(a)-x_{hq}(a) \leqslant 2^q(c(a)-x_h(a)) +F_{h-2}$$

when h is even.

Proof: this lemma is verified for h = 4. As mentionned before, the augmenting path P_{h+1} $(h+1 \leqslant k)$ augments the flow on the supplementary arcs with initial capacity S_ℓ, $\ell \equiv h+2 \pmod 3$ by an amount equal to the flow on the special arc a_i, $i \equiv h \pmod 3$. Hence, for h odd:

$$2^q(c(a)-x_h(a))-F_{h-2}-1-2^q r^{h-1}-F_{h-1} \leqslant C_q\ (a)-x_{h+1,q}(a) \leqslant 2^q(c(a)-x_h(a))+F_{h-1} +2^q r^{h-1}$$

that is $2^q(c(a)-x_{h+1}(a))-F_h-1 \leqslant c_q(a)-x_{h+1,q}(a) \leqslant 2^q(c(a)-x_{h+1}(a))+F_{h-1}$

for the supplementary arcs on P_{h+1}. Since the flow has not been modified on the other supplementary arcs, the lemma is verified for h+1 (which is even) and all supplementary arcs. The proof for h even is similar. #

In the augmenting network $N_{x_{hq}}$ associated to N^q and the flow x_{hq}, the capacity $c_{x_{hq}}(a)$ of any arc a is bounded by

$$2^q c_{x_h}(a)-F_{h-1}-1 \leqslant c_{x_{hq}}(a) \leqslant 2^q c_{x_h}(a)+F_{h-1}+1$$

where $c_{x_h}(a)$ is the capacity of the corresponding arc in N_{x_h}.

The relation extends to augmenting paths P:

$$2^q c_{x_h}(P)-F_{h-1}-1 \leqslant c_{x_{hq}}(P) \leqslant 2^q c_{x_h}(P)+F_{h-1}+1$$

for all integer h, $4 \leqslant h \leqslant k$.

Let P_{k+1} be the unique augmenting path which is selected by Capacity in N. By considering the flows given in table 1 and using the relations between S_1, S_2, S_3 and the powers of r, one obtains:

$$c_{x_k}(P_{k+1}) \geqslant c_{x_k}(P)+r^{k-2}(1-S_1)$$

for all augmenting path $P \neq P_{k+1}$.

It follows that

$$C_{x_{kq}}(P_{k+1}) \geqslant C_{x_{kq}}(P) + 2^q r^{k-2}(1-S_1)-2F_{k-1}-2$$

Since we assume that k is the last iteration such that P_{k+1} is selected, it must be true that

$$2F_{k-1}+2-2^q r^{k-2}(1-S_1) \geqslant 0.$$

Using the well-know relation

$$F_n \leqslant (1+r)^{n-1} \qquad \text{all } n \geqslant 1$$

where $1+r = \frac{1}{2}(\sqrt{5}+1)$ is the "golden ratio", one obtains:

$$2(1+r)^{k-2}+2 \geqslant 2^q r^{k-2}(1-S_1)$$

$$\text{so} \quad \left(\frac{1+r}{r}\right)^{k-2} + \frac{1}{r^{k-2}} \geqslant 2^{q-1}(1-S_1)$$

Using the relation $\frac{1}{r} < \frac{1+r}{r}$, it follows

$$\left(\frac{1+r}{r}\right)^{k-2} \geqslant 2^{q-2}(1-S_1)$$

that is

$$k \geqslant q\,\frac{\log 2}{\log(1+1/r)} - \frac{2\log 2 - \log(1-S_1)}{\log(1+1/r)} - 2$$

or $\quad k \geqslant O(q)$ #

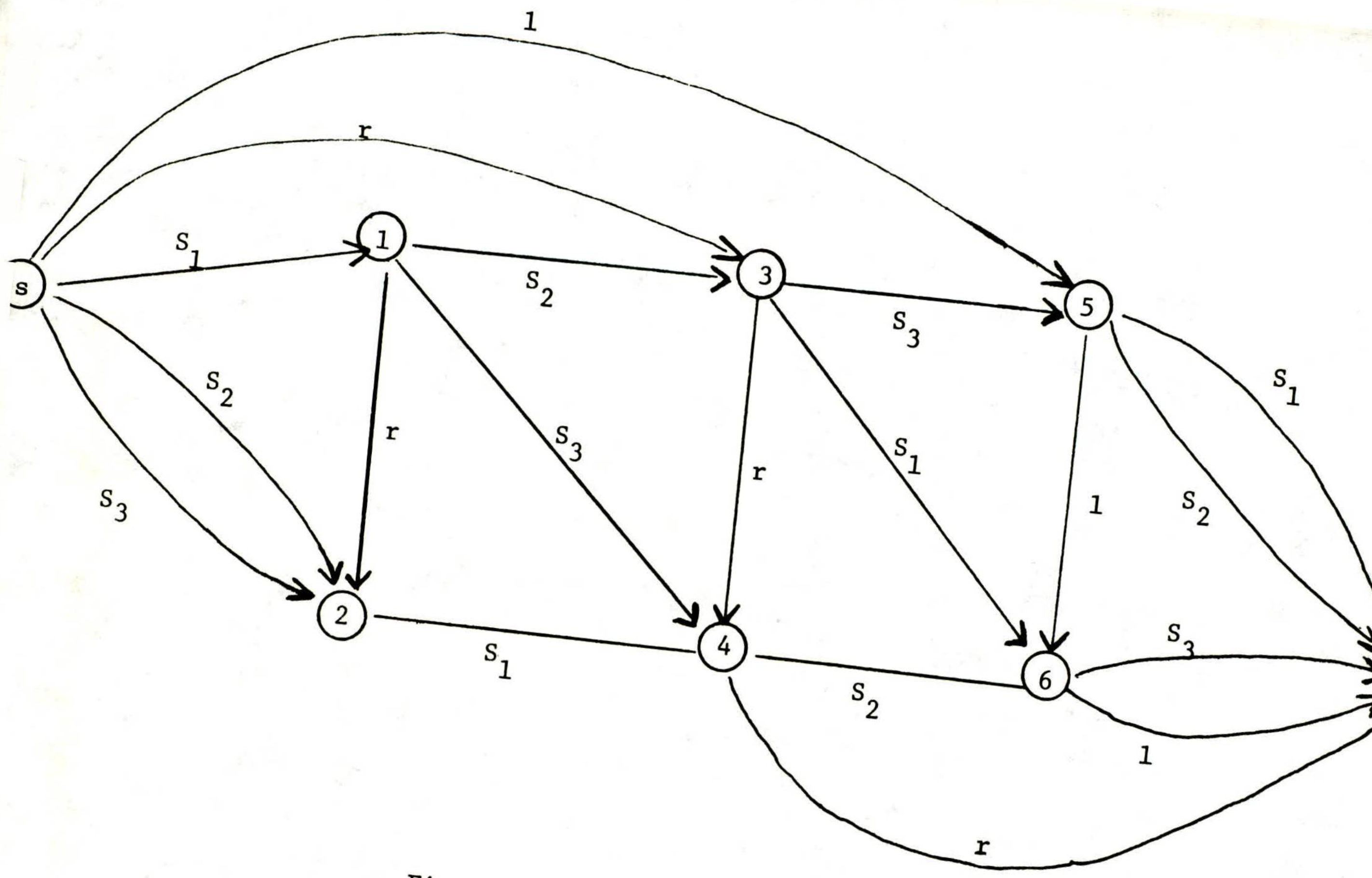

Figure 1: A bad network for Capacity.