

AD-A097 761 CARNEGIE-MELLON UNIV PITTSBURGH PA MANAGEMENT SCIENC--ETC F/G 12/1
WORST CASE ANALYSIS OF A CLASS OF SET COVERING HEURISTICS.(U)
JUN 79 A C HO
UNCLASSIFIED MSRR-473 NL

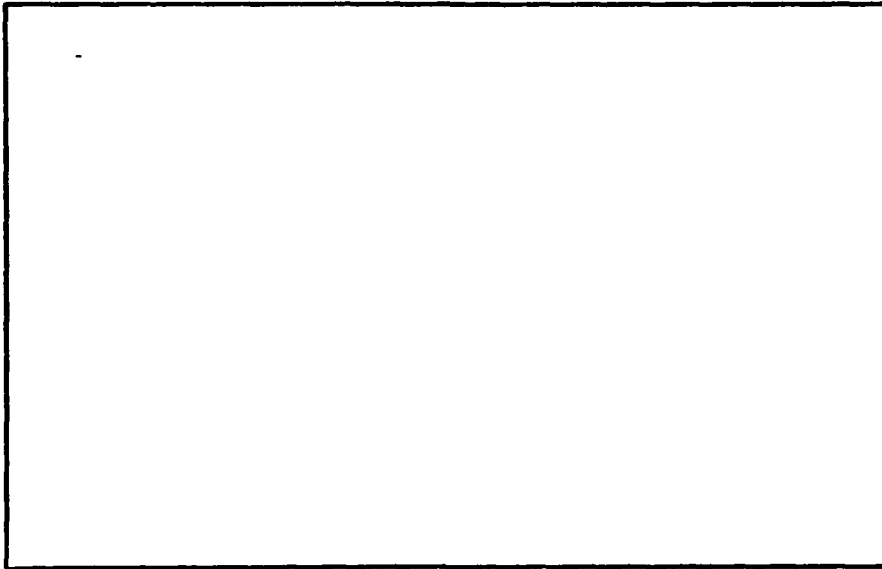
For
A

END
DATE
FILMED
9-80
DTIC

LEVEL II

①
B 2

AD A 097 761



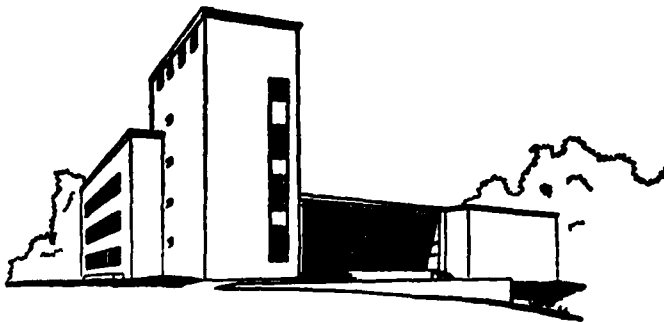
Carnegie-Mellon University

PITTSBURGH, PENNSYLVANIA 15213

**DTIC
ELECTED
APR 15 1981**
C

GRADUATE SCHOOL OF INDUSTRIAL ADMINISTRATION

WILLIAM LARIMER MELLON, FOUNDER



DISTRIBUTION STATEMENT A
Approved for public release;
Distribution is unlimited

81 4 13

88

DTIC FILE COPY

①
④ Management Sciences Research Report No. 473 ✓

⑥
14 MSRI-912
WORST CASE ANALYSIS OF A CLASS OF
SET COVERING HEURISTICS.

by

⑦ Andrew C. /Ho

DTIC
APR 15 1981

⑧ Jun 79 12 21

This report was prepared as part of the activities of the Management Sciences Research Group, Carnegie-Mellon University. Reproduction in whole or in part is permitted for any purpose of the U.S. Government.

Management Science Research Group
Graduate School of Industrial Administration
Carnegie-Mellon University
Pittsburgh, Pennsylvania 15213

DISTRIBUTION STATEMENT A

Approved for public release;
Distribution Unlimited

40 2426

Abstract

In [2], Chvatal provided the tight worst case bound of the set covering greedy heuristic. We considered a general class of set covering heuristics. Their worst case bounds are dominated by that of the greedy heuristic.

Accession For	
NTIS GRA&I	☒
DTIC TAB	☐
Unannounced	☐
Justification	
By _____	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A	

1. Introduction

→ The Set Covering problem is notoriously hard to solve and is, in fact, NP-complete. [1] A good heuristic algorithm that gives a close approximation to the optimum is therefore desirable. In [2], Chvatal found the tight worst case bound of the greedy heuristic commonly considered in the literature. In this paper, we investigate the worst case behavior of a general class of heuristic algorithms. These worst case bounds are found to be dominated by that of the greedy heuristic.

We consider the Set Covering problem

$$(1) \quad \text{Min } \{cx \mid Ax \geq e, x \text{ binary}\}$$

where $A = \langle a_{ij} \rangle$ is $m \times n$ with $a_{ij} = 0, 1$ for all i, j ; $e = (1, \dots, 1)^T$ is $m \times 1$; x is $n \times 1$ and $c \in \mathbb{R}^n$ is $1 \times n$. For notation purposes, we define

$M = \{1, \dots, m\}$ as the set of row indices,

$N = \{1, \dots, n\}$ as the set of column indices,

$M_j = \{i \in M \mid a_{ij} = 1\}$ for every $j \in N$

and $N_i = \{j \in N \mid a_{ij} = 1\}$ for every $i \in M$.

Any feasible solution is said to be a cover. Any nonredundant cover is said to be prime. If $x_j = 1$ in a feasible solution to (1), variable j is said to cover all rows $i \in M_j$. Without loss of generality, we assume

$$(2) \quad \begin{aligned} c_j &> 0 & \text{all } j \in N \\ M_j &\neq \emptyset & \text{all } j \in N \\ N_i &\neq \emptyset & \text{all } i \in M. \end{aligned}$$

The worst case performance is measured by the smallest bound Q on the ratio $Z_{\text{heu}}/Z_{\text{opt}}$, i.e.

$$\frac{Z_{\text{heu}}}{Z_{\text{opt}}} \leq Q$$

where Z_{heu} and Z_{opt} are the values of the heuristic and optimal solutions. Due to our assumptions in (2), there exists at least one feasible solution and $Z_{\text{opt}} > 0$ holds. The ratio $Z_{\text{heu}}/Z_{\text{opt}}$ is well-defined.

2. Algorithm I

The class of heuristic algorithms that we consider is a generalization of the greedy heuristic. In essence, the heuristic sets a value of one variable at a time until a cover is found. Each variable is evaluated according to its cost and the number of rows that it may cover. We let R_r be the set of uncovered rows before the r th variable is chosen by the heuristic, $S(x)$ be the support of the cover to be found and k_{rj} be the number of additional rows variable j can cover. We call this class of heuristics Algorithm I.

Step 0 Let $R_1 = M$, $S(x) = \emptyset$ and $r = 1$. Go to 1.

Step 1 If $R_r = \emptyset$, go to 2.

Otherwise, define $k_{rj} = |M_j \cap R_r|$ for all $j \in N$. Let $j^* \in N$ be such that

$$f(c_{j^*}, k_{rj^*}) = \min_{j \in N} f(c_j, k_{rj}).$$

In case of a tie, a fixed but arbitrary tie breaking rule is used.

Set

$$S(x) \leftarrow S(x) \cup \{j^*\}$$

$$R_{r+1} \leftarrow R_r \setminus M_{j^*}$$

$$r \leftarrow r+1$$

and go to 1.

Step 2 Let $x_j = 1$ $j \in S(x)$
 $= 0$ otherwise

and stop.

A function f is used to evaluate the variables. A different function used will correspond to a different heuristic. For obvious reasons, we require

$$f(c_j, 0) \leq +\infty.$$

Otherwise, we consider any $f : R^+ \times Z^+ \rightarrow R$ where R^+ is the set of positive real numbers representing c_j and Z^+ is the set of positive integers representing k_{rj} .

The greedy heuristic that Chvatal considered in [2] is a special case of Algorithm I when $f(c_j, k_{rj}) = c_j/k_{rj}$. The tight worst case bound that Chvatal derived is

$$(3) \quad \frac{Z_{\text{heu}}}{Z_{\text{opt}}} \leq H(d)$$

$$\text{where} \quad H(d) = \sum_{j=1}^d \frac{1}{j}$$

$$\text{and} \quad d = \max_{j \in N} |M_j|.$$

This bound is dependent on problem size, density and the distribution of the nonzero coefficients in the matrix A . The function $H(d)$ is, in turn, bounded by $\log d$ for reasonably large d . Similar results for special classes of problems were obtained previously by Johnson [4] and Lovasz [6].

Any fixed but arbitrary tie breaking rule may be used. The tie breaker may use any data that is available, including c_j and k_{rj} . Without loss of generality, we assume that the tie breaking rule is different from the function f used so that if there exist $j_1, j_2 \in N$ with $j_1 \neq j_2$, $k_{rj_1} \neq k_{rj_2}$ but $f(c_{j_1}, k_{rj_1}) = f(c_{j_2}, k_{rj_2})$, the tie breaker will break the tie. When all rules fail, we allow breaking ties arbitrarily by the location of ones so that a variable can always be chosen. A good example will be to choose j_1 if $j_1 < j_2$.

In the next theorem, we show that the worst case performance of any heuristic in Algorithm I is dominated by that of the greedy heuristic. We also use the symbol $\leq$, when used in

$$f(c_{j_1}, k_{rj_1}) \leq f(c_{j_2}, k_{rj_2}),$$

to indicate either

$$f(c_{j_1}, k_{rj_1}) < f(c_{j_2}, k_{rj_2})$$

or

$$f(c_{j_1}, k_{rj_1}) = f(c_{j_2}, k_{rj_2}) \text{ but the tie breaker chooses } j_1.$$

Theorem 1

Assume Algorithm I is used. There is no function f that gives a worst case bound strictly better than $H(d)$ for any $d \geq 1$.

Proof

By contradiction. Notice that the theorem is trivial when $d=1$ as $Z_{\text{heu}} \geq Z_{\text{opt}}$ implies $Z_{\text{heu}}/Z_{\text{opt}} \geq H(1)$. We assume f is a function, when used in Algorithm I, that gives

$$(4) \quad \frac{Z_{\text{heu}}}{Z_{\text{opt}}} \leq Q_d < H(d) \quad \text{for some } d \geq 2.$$

We consider two cases.

Case 1 We assume, for all $d \geq 2$ and $a > 0$,

$$(5) \quad f\left(\frac{ad}{j}, d\right) \leq f(a, j) \quad \text{all } j=1, \dots, d-1.$$

Let $k_j \in \{1, \dots, j\}$ for $j=1, \dots, d$ be such that

$$(6) \quad f\left(\frac{ad}{k_j}, d\right) \leq f\left(\frac{ad}{k}, d\right) \quad k=1, \dots, j.$$

Consider the problem

$$\begin{aligned} \text{Min } & \sum_{j=1}^d \frac{ad}{k_j} x_j + \sum_{i=1}^d a y_i \\ & x_j + y_i \geq 1 \quad i=1, \dots, d \\ & x_j, y_i = 0 \text{ or } 1 \quad j=1, \dots, d \end{aligned}$$

From (6), x_d is chosen over $x_1, \dots, x_{d-1}$. If $k_d \neq d$, x_d is also chosen over $y_1, \dots, y_d$. If $k_d = d$, x_d is identical to $y_1, \dots, y_d$ except for the location of ones in the matrix. The tie breaker will fail but we can always rearrange the matrix so that x_d can be chosen arbitrarily. In either case, x_d is chosen first. From (5) and (6), $x_{d-1}, \dots, x_1$ are then chosen sequentially for the solution $x_j = 1, y_i = 0$ for $i, j = 1, \dots, d$.

The optimal solution is $x_j = 0, y_i = 1$ for $i, j=1, \dots, d$ with

$$\begin{aligned} \frac{Z_{\text{heu}}}{Z_{\text{opt}}} &= \frac{\sum_{j=1}^d \frac{ad}{k_j}}{ad} \\ &\geq \sum_{j=1}^d \frac{1}{j} \quad \text{as } k_j \leq j. \end{aligned}$$

which contradicts with (4).

Case 2 (5) does not hold for some $d \geq 2$.

Without loss of generality, let d be the smallest integer so that (5) does not hold. We prove by induction on d .

Subcase 2.1 $d = 2$

The negation of (5) gives

$$(7) \quad f(a, 1) \leq f(2a, 2) \quad \text{some } a > 0.$$

$$(8) \quad \text{Let } c = a \quad \text{if } f(a, 1) \leq f(2a, 1) \\ = 2a \quad \text{otherwise}$$

and consider

$$\text{Min } cx_1 + 2a x_2 + 2a x_3$$

s.t.

$$x_j + x_3 \geq 1 \quad j = 1, 2$$

$$x_j = 0, 1 \quad j = 1, 2, 3$$

From (7) and (8), the heuristic chooses x_1 first. Then, regardless of which variable the heuristic chooses next, we have $Z_{\text{heu}} = 2a + c$. The optimal solution is $x_1 = x_2 = 0, x_3 = 1$ with

$$\frac{Z_{\text{heu}}}{Z_{\text{opt}}} = \frac{2a+c}{2a} \geq \frac{3a}{2a} = H(2)$$

which contradicts with (4).

Subcase 2.2 $d \geq 3$.

Since (5) holds for $d-1$ but not for d , we have

$$(9) \quad f\left(\frac{ad}{p}, d\right) \geq f(a, p) \quad \text{some } a > 0, \text{ some } p \in \{1, \dots, d-1\}$$

and

$$(10) \quad f\left(\frac{c(d-1)}{j}, d-1\right) \leq f(c, j) \quad \text{all } c > 0, j=1, \dots, d-2$$

Claim $f\left(\frac{ad}{p}, d\right) \geq f\left(\frac{a(d-1)}{p}, d-1\right)$. If $p = d-1$ in (9), it is trivial. If $p < d-1$, (9) and using $j = p$, $c = a$ in (10) give

$$f\left(\frac{ad}{p}, d\right) \geq f(a, p) \geq f\left(\frac{a(d-1)}{p}, d-1\right).$$

Let $c = \frac{ad}{p}$ and (10) can be generalized to

$$(11) \quad f\left(\frac{c(d-1)}{j}, d-1\right) \leq f(c, j) \quad j=1, \dots, d-2, d$$

Let $k_j \in \{1, \dots, j\}$ for all $j=1, \dots, d$ be such that

$$(12) \quad f\left(\frac{c(d-1)}{k_j}, d-1\right) \leq f\left(\frac{c(d-1)}{k}, d-1\right) \quad k=1, \dots, j$$

Consider the problem

$$\begin{aligned} \text{Min } & \sum_{j=1}^d \frac{c(d-1)}{k_j} x_j + \sum_{i=1}^{d-1} c y_i \\ \text{s.t. } & \end{aligned}$$

$$x_j + y_i \geq 1 \quad \begin{array}{l} i=1, \dots, d-1, j=1, \dots, d \\ \text{except for } i=1, j=d \end{array}$$

$$\lambda x_{d-1} + x_d + y_1 \geq 1$$

$$x_j, y_i = 0 \text{ or } 1 \quad i=1, \dots, d-1, j=1, \dots, d$$

where

$$\begin{aligned} \lambda &= 1 & \text{if } k_{d-1} &= d-1 \\ &= 0 & \text{otherwise} \end{aligned}$$

2.2.1 $\lambda = 0$ and $k_{d-1} < d-1$

From (11) and (12), x_d is chosen first. From (12), x_{d-1} is then chosen over $x, \dots, x_{2-2}$. Since all y_i 's have $d-1$ ones left and $k_{d-1} < d-1$,

$$f\left(\frac{c(d-1)}{k_{d-1}}, d-1\right) \leq f\left(\frac{c(d-1)}{d-1}, d-1\right) = f(c, d-1)$$

and x_{d-1} is chosen over $y_1, \dots, y_{d-1}$. From (11) and (12), the heuristic will then choose $x_{d-2}, \dots, x_1$ to complete the solution.

2.2.2

From (12), x_d is chosen over $x_1, \dots, x_{d-2}$ first. As x_{d-1} has cost c , x_{d-1} is identical to all y_i 's. From (11) and (12), x_d is also chosen over $x_{d-1}, y_1, \dots, y_{d-1}$. From (12), x_{d-1} is chosen over $x_1, \dots, x_{d-2}$. Since x_{d-1} is identical to all y_i 's except for the location of ones, all tie breaking rules will fail but we can always rearrange the matrix so that x_{d-1} is chosen arbitrarily. From (11) and (12), the heuristic will choose $x_{d-2}, \dots, x_1$ sequentially.

In either case, the heuristic solution is $x_j = 1, y_i = 0$. $j = 1, \dots, d, i = 1, \dots, d-1$. The optimal solution is $x_j = 0, y_i = 1$ for all $i = 1, \dots, d-1, j = 1, \dots, d$ with

$$\frac{z_{\text{heu}}}{z_{\text{opt}}} = \frac{\sum_{j=1}^d \frac{c(d-1)}{k_j}}{c(d-1)} \geq \sum_{j=1}^d \frac{1}{j} \quad \text{as } k_j \leq j$$

which contradicts with (4).

Q.E.D.

The cover obtained from Algorithm I is not necessarily prime. It is possible to implement a simple procedure to derive a prime cover from the heuristic solution. See, for example, [1], [3]. The value of the prime cover and consequently the worst case behavior may improve. The next theorem shows that, with some general assumption on the tie breaking rule, the result of Theorem 1 still holds.

We assume that, if a tie exists, the tie breaking rule is based on c_j and k_{rj} only so that the tie breaker will work whenever we have two variables $j_1 \neq j_2$ with $c_{j_1} \neq c_{j_2}$. In case of a second tie, we allow breaking ties utilizing additional data that is available, including breaking ties arbitrarily.

Theorem 2

Assume a procedure is used to strengthen a solution from Algorithm I to prime. Assume further that the tie breaker is as previously described. There is no function f that gives a worst case bound strictly better than $H(d)$.

Proof

It suffices to show that all heuristic solutions are prime. Notice first that, with the changes in the tie breaker, Algorithm I will give the same solutions for all counter examples in the proof of Theorem 1. Since the heuristic solutions are prime except for subcase 2.1, it suffices to consider subcase 2.1 only. We have $d=2$ and

$$(13) \quad f(a,1) \leq f(2a, 2) \text{ for some } a > 0.$$

Let $\Delta = \text{Min } (a, 2a (H(2) - Q_2))$ and consider

$$\text{Min } \sum_{j=1}^3 c_j x_j$$

s.t.

$$x_j + x_3 \leq 1 \quad j=1, 2$$

$$x_j = 0, 1 \quad j=1, 2, 3$$

Case 1 There exists $\theta \in (-\Delta, \frac{\Delta}{Q_2})$ such that

$$(14) \quad f(2a+\theta, 1) \leq f(2a, 1).$$

Let $c_1 = a$ if $f(a, 1) \leq f(2a + \theta, 1)$

$= 2a + \theta$ otherwise

$$c_2 = 2a + \theta$$

$$c_3 = a.$$

From (13), the heuristic chooses x_1 first. From (14), x_2 is picked next for the prime cover $x_1 = x_2 = 1, x_3 = 0$. The optimal solution is $x_1 = x_2 = 0, x_3 = 1$ with

$$\frac{Z_{\text{heu}}}{Z_{\text{opt}}} \geq \frac{3a+\theta}{2a} > H(2) - \frac{\Delta}{2a} \geq Q_2$$

which contradicts with (4).

Case 2 For all $\theta \in (-\Delta, \frac{\Delta}{Q_2})$

$$(15) \quad f(2a + \theta, 1) \geq f(2a, 1)$$

Subcase 2.1 There exists $\theta \in (-\Delta, \frac{\Delta}{Q_2})$ so that

$$(13') \quad f(2a + \theta, 2) \geq f(a, 1)$$

Let $c_1 = a$ if $f(a, 1) \leq f(2a, 1)$

$= 2a$ otherwise

$$c_2 = 2a$$

$$c_3 = 2a + \theta$$

From (13') and then (15), the heuristic chooses x_1 and then x_2 for the prime cover $x_1 = x_2 = 1, x_3 = 0$. The optimal solution is $x_1 = x_2 = 0, x_3 = 1$ with

$$\frac{Z_{heu}}{Z_{opt}} = \frac{3a}{2a+\theta} > \frac{3a}{2a + \frac{\Delta}{Q_2}}$$

$$\geq \frac{H(2)}{1 + \frac{H(2)-Q_2}{Q_2}}$$

$$= Q_2$$

which contradicts with (4).

Subcase 2.2 $f(2a+\theta, 2) \leq f(a,1)$

This is case 1 in the proof of theorem 1.

Consider

$$\text{Min } (2a+\theta) x_1 + ax_2 + ay_1 + ay_2$$

s.t.

$$x_j + y_i \geq 1 \quad i=1,2, j=1,2$$

$$x_j, y_i = 0,1$$

The heuristic chooses x_2 arbitrarily and then x_1 for the prime cover

$x_1 = x_2 = 1, y_1 = y_2 = 0$. The optimal solution is $y_1 = y_2 = 1,$

$x_1 = x_2 = 0$ with

$$\frac{Z_{heu}}{Z_{opt}} = \frac{3a+\theta}{2a} > H(2) - \frac{\Delta}{2a} = Q_2$$

which contradicts with (4).

Q.E.D.

3. Extensions of Algorithm I: Algorithm II.

The worst case bound for Algorithm I is dependent on problem size. We are then interested to find other heuristic algorithms that may give a better worst case performance. Algorithm II is an extension of

Algorithm I in that it chooses one variable at a time. The difference is in step one where the variable is chosen only from a subset of all variables that are available. More specifically, a row is chosen first and the heuristic then chooses a variable with a nonzero coefficient in that row. Algorithm II is computationally more efficient and since all rows must be covered eventually, Algorithm II chooses one variable to cover the row that is considered most essential first.

Step 0 Let $R_1 = M$, $S(x) = 0$, $r = 1$ and go to 1.

Step 1 If $R_r = 0$, go to 2.

Otherwise, define $k_{rj} = |M_j \cap R_r|$.

(a) Pick $i_r \in R_r$.

(b) Pick $j_r^* \in N_{i_r}$ so that

$$f(c_{j_r^*}, k_{rj_r^*}) = \min_{j \in N_{i_r}} f(c_j, k_{rj})$$

Set $S(x) \leftarrow S(x) \cup \{j_r^*\}$

$$R_{r+1} \leftarrow R_r \setminus M_{j_r^*}$$

$$r \leftarrow r+1$$

and go to 1.

Step 2 Let $x_j = 1 \quad j \in S(x)$
 $= 0 \quad \text{otherwise}$

Different rules can be used to pick the row i_r in step 1a.

A different rule will correspond to a different class of heuristic algorithms.

Theorem 3

Assume $f(c_j, k_{rj}) = c_r/k_{rj}$. Regardless of the rule used in step 1a in Algorithm II, the worst case bound is

$$\frac{Z_{\text{heu}}}{Z_{\text{opt}}} \leq dH(d)$$

where $H(d) = \sum_{j=1}^d \frac{1}{j}$ and $d = \max_{j \in N} |M_j|$.

Proof

Let x^* and $\bar{x}$ be the optimal and heuristic solutions respectively and let $S(x) = \{j \in N | x_j = 1\}$. Since x^* must be feasible, it covers the rows i_r , $r = 1, \dots, k$ where $k = |S(\bar{x})|$. Then, there exists at least one $j(r) \in S(x^*)$ so that $j(r) \in N_{i_r}$ for every r . We have

$$\frac{c_{j_r^*}}{k_{rj_r^*}} \leq \frac{c_{j(r)}}{k_{rj(r)}} \quad r=1, \dots, k$$

and

$$\begin{aligned} Z_{\text{heu}} = c\bar{x} &= \sum_{r=1}^k c_{j_r^*} \leq \sum_{r=1}^k c_{j(r)} \frac{k_{rj_r^*}}{k_{rj(r)}} \\ &= \sum_{j \in S(x^*)} c_j \left(\sum_{r \in S_j} \frac{k_{rj_r^*}}{k_{rj}} \right) \end{aligned}$$

where $S_j = \{r | j(r) = j\}$.

$k_{rj_r^*} \leq |M_{j_r^*}| \leq d$ implies $Z_{\text{heu}} \leq d \sum_{j \in S(x^*)} c_j \left(\sum_{r \in S_j} \frac{1}{k_{rj}} \right)$.
 Claim that $\sum_{r \in S_j} \frac{1}{k_{rj}} \leq \sum_{j=1}^d \frac{1}{j}$ for every $j \in S(x^*)$. For $j \in S(x^*)$ such that $|S_j| \leq 1$, $\sum_{r \in S_j} \frac{1}{k_{rj}} \leq \sum_{j=1}^d \frac{1}{j}$ is trivial. Suffice to consider $j \in S(x^*)$ such

that $|S_j| \geq 2$. Let $r_1, r_2 \in S_j$ with $r_1 \neq r_2$. Without loss of generality, $r_1 < r_2$. From the definitions of S_j and $j(r)$, $i_{r_1}, i_{r_2} \in M_j$. Since

$$i_{r_1} \notin R_{r_2}, k_{r_1 j} = |M_j \cap R_{r_1}| \geq |M_j \cap R_{r_2}| - 1 = k_{r_2 j} - 1.$$

In general, $k_{r_1 j} \neq k_{r_2 j}$ for all $r_1, r_2 \in S_j$ and $r_1 \neq r_2$. As

$$k_{r j} \in \{1, \dots, |M_j|\},$$

$$\sum_{r \in S_j} \frac{1}{k_{r j}} \leq \frac{|M_j|}{\sum_{j=1}^d} \frac{1}{j} \leq \sum_{j=1}^d \frac{1}{j} = H(d).$$

Substitution yields

$$\begin{aligned} Z_{\text{heu}} &\leq d \sum_{j \in S(x^*)} c_j \left(\sum_{r \in S_j} \frac{1}{k_{r j}} \right) \\ &\leq dH(d) \sum_{j \in S(x^*)} c_j \\ &= dH(d) Z_{\text{opt}} \end{aligned}$$

Q.E.D.

Different rules can be used in Step 1a. Two specific rules are considered here. In the first rule, we pick a row of minimum cardinality so that a row with fewer potential candidates to be chosen from is covered first. We call it Algorithm II.1. In the second rule, a penalty for choosing a wrong variable is computed. The row with the largest penalty is chosen first. The penalty for every row is the difference between the two smallest functional values. We call this Algorithm II.2. The details are outlined as follows.

Algorithm II.1

$$\text{Step 1a} \quad |N_{i_r}| = \min_{i \in R_r} |N_i|$$

Algorithm II.1

Step 1a

(i) For $i \in R_r$ such that $|N_i| = 1$, define $P(i) = +\infty$.

(ii) For $i \in R_r$ such that $|N_i| \geq 2$, define $j_1^i, j_2^i \in N_i$ such that

$$f(c_{j_1^i, k_{rj_1^i}}) \leq f(c_{j_2^i, k_{rj_2^i}}) \leq f(c_j, k_{rj}) \quad j \in N_i \setminus \{j_1^i, j_2^i\}$$

$$\text{and } P(i) = f(c_{j_2^i, k_{rj_2^i}}) - f(c_{j_1^i, k_{rj_1^i}}).$$

Pick i_r by

$$P(i_r) = \max_{i \in R_r} P(i)$$

Remark 4 If $f(c_j, k_{rj}) = c_j/k_{rj}$ is used in either Algorithm II.1 or II.2, the bound $dH(d)$ is tight.

Proof Let $\frac{Z_{\text{heu}}}{Z_{\text{opt}}} \leq Q_d < dH(d)$. Consider, for any $d \geq 2$,

$$c > \epsilon > 2\delta > 0, \quad \delta < \frac{c(dH(d) - Q_d)}{2Q_d},$$

$$\begin{array}{ll} \text{Min} & \sum_{j=1}^d \frac{cd}{j} x_j + (c+\delta)x_{d+1} + \sum_{j=1}^{d+2} (c+\epsilon)y_j + \sum_{j=1}^{d+2} \delta z_j \\ \text{s.t.} & \end{array}$$

$$x_j + x_{d+1} + \sum_{i=j}^d y_i \geq 1 \quad j=1, \dots, d$$

$$x_j + \sum_{i=1}^{d+2} z_i \geq 1$$

$$x_j, y_j, z_j = 0, 1$$

The rows and columns chosen are $d, d-1, \dots, 1$ and $x_d, x_{d-1}, \dots, x_1$ in that order. An optimal solution is $x_j=0, y_j=0 \quad j=1, \dots, d, x_{d+1}=1, z_1=1, z_j=0 \quad j=2, \dots, d+2$.

We have

$$\frac{Z_{\text{heu}}}{Z_{\text{opt}}} = \frac{\sum_{j=1}^d \frac{cd}{j}}{c+2\delta} = \frac{cdH(d)}{c+2\delta} > \frac{dH(d)}{1 + \frac{dH(d)-Q_d}{Q_d}} = Q_d$$

for the contradiction.

Theorem 5

Assume either Algorithm II.1 or II.2 is used. There is no function f that gives a worst case bound strictly better than $H(d)$, for any $d \geq 1$.

Proof

The proof is the same as that of Theorem 1. The heuristic will choose a sequence of rows such that the same variables and, hence, the same solution in all counter examples are chosen. Notice that $|N_i|$ is the same for all rows and if Algorithm II.1 is used, the rows can be chosen arbitrarily for the desired result.

Q.E.D.

3. Conclusion

Two general classes of heuristic algorithms were considered. They are easy to implement as the variables are evaluated essentially on the cost and the number of rows that can be covered. The worst case performances of all heuristics are dominated by the function $H(d)$ which, in turn, is bound by $\log d$ and is dependent on problem size and distribution. A different approach would be needed in order to find a heuristic that gives a better worst case bound. From the proofs, the worst case bounds for different functions and, hence, heuristics are attained in different examples. A combination of some functions may improve the average performance. A computational study is available in [1].

Acknowledgment

The author would like to thank Professors Egon Balas and Gerald Cornuejols, Graduate School of Industrial Administration, Carnegie-Mellon University, for their help and encouragement.

References

1. E. Balas and A. Ho, "Set covering algorithm using cutting planes, heuristics and subgradient optimization: A computational study", Mathematical Programming Study 12 (1980) 37-60.
2. V. Chvatal, "A greedy heuristic for the set covering problem", Publication 284, Departement d' informatique et de recherche operationnelle, Universite de Montreal, 1978.
3. R. Garfinkel and G. Nemhauser, "Integer Programming", John Wiley (New York) 1972.
4. D. Johnson, "Approximation algorithms for combinatorial problems", Journal of Computer and System Science 9 (1974) 256-278.
5. R. Karp, "Reducibility among combinatorial problems", in: R. Miller and J. Thatcher, ed., Complexity of Computer Computations (Plenum Press, New York, 1972) 85-103.
6. L. Lovasz, "On the ratio of optimal integral and fractional covers", Discrete Mathematics 13 (1975) 383-390.

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER Technical Report No. 473	2. GOVT ACCESSION NO. AD-A097761	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Worst Case Analysis of a Class of Set Covering Heuristics	5. TYPE OF REPORT & PERIOD COVERED Technical Report June 1979	
	6. PERFORMING ORG. REPORT NUMBER M.S.R.R. No. 473	
7. AUTHOR(s) Andrew C. Ho	8. CONTRACT OR GRANT NUMBER(s)	
9. PERFORMING ORGANIZATION NAME AND ADDRESS Graduate School of Industrial Administration Carnegie-Mellon University Pittsburgh, Pennsylvania 15213	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS	
11. CONTROLLING OFFICE NAME AND ADDRESS Personnel and Training Research Programs Office of Naval Research (Code 434) Arlington, Virginia 22217	12. REPORT DATE June 1979	
	13. NUMBER OF PAGES 16	
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report)	
	16. DECLASSIFICATION/DOWNGRADING SCHEDULE	
17. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		
18. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
19. SUPPLEMENTARY NOTES		
20. KEY WORDS (Continue on reverse side if necessary and identify by block number) Set covering, heuristic algorithms, worst case analysis, bounds		
21. ABSTRACT (Continue on reverse side if necessary and identify by block number) In [2], Chvatal provided the tight worst case-bound of the set covering greedy heuristic. We considered a general class of set covering heuristics. Their worst case bounds are dominated by that of the greedy heuristic.		

DD FORM 1473

EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-914-6601

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)