アルゴリズム 47－3
（1995． 9． 21）

最小キーおよび最適部分構造スクリーンの近似

阿久津 達也　　鮑 豊
群馬大学 工学部 情報工学科

化学構造データベースでは検索の効率化のために部分構造スクリーンが用いられるが、本稿ではスクリーンとして使用される最適な部分構造を選択する問題について考察する。この問題は関係データベース理論における最小キー問題と密接に関連しているが、両者ともに NP 困難であるので、多項式時間近似アルゴリズムの観点から議論を行う。本稿では、両者ともに近似度は SET COVER 問題と同等 ($\Theta(\log n)$) であること、Kolmogorov 計算量を応用することにより部分構造スクリーン問題が平均的には定数以内の近似が可能であること、などを示す。

Approximating Minimum Keys and Optimal Substructure Screens

Tatsuya Akutsu　and　Feng Bao
Department of Computer Science, Gunma University
1-5-1 Tenjin, Kiryu, Gunma 376 Japan
e-mail: akutsu@cs.gunma-u.ac.jp bao@comp.cs.gunma-u.ac.jp

In this paper, we consider the problem of selecting an optimal substructure screen, which is important in database systems for chemistry. This problem is closely related to the minimum cardinality key problem. Since both problems are NP-hard, we consider polynomial time approximation algorithms. We show that the performance ratios of both problems are $\Theta(\log n)$ using reductions from/to the SET COVER problem. On the other hand, using Kolmogorov complexity, we show that the optimal substructure screen problem can be approximated within a factor of $2 + \epsilon$ in the average case.

1 Introduction

In chemistry, database management systems for chemical structures are very important since the number of known chemical structures is very large. In particular, the following database search problem (*substructure search*) is very important [1, 15]: given a part of chemical structure S, enumerate all chemical structures which contain substructures isomorphic to S. For rapid substructure search, *substructure screens* have been used in several practical database systems [1, 15].

Substructure screens are defined and used as follows (see Fig .1). For each chemical structure G, a bit vector $b(G)$ is associated. If G contains a substructure isomorphic to a fixed structure B_i, then i-th bit $b(G)|i$ of $b(G)$ is 1, otherwise 0. Note that if $b(S)|i = 1$ and $b(G)|i = 0$ for some i, G must not contain a substructure isomorphic to S and we need not test G. Therefore these bit vectors can be used as screens before testing whether or not G

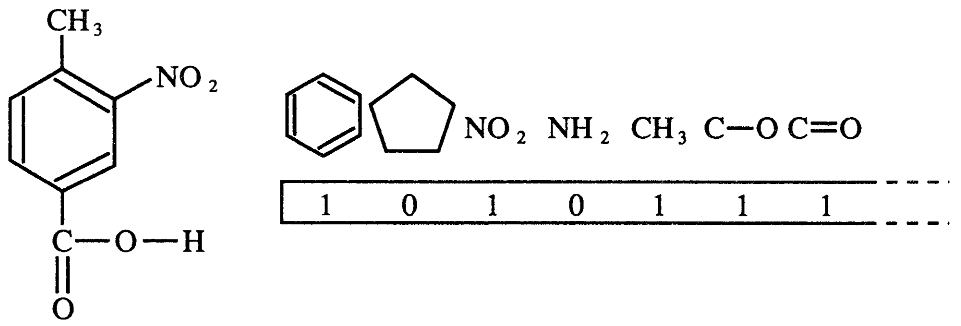

Figure 1: Example of a substructure screen.

contains a subgraph isomorphic to S. Moreover index files can be constructed and utilized from these bit vectors.

Although screens have been utilized effectively in existing database systems, no studies have been done about how to select substructures used as a screen. Thus we have studied such a method in this paper. For that purpose, the following approach seems reasonable: a large set of candidate substructures is tested, and then a subset is selected which minimizes the maximum number of structures in a database having the same bit vectors. Moreover two approaches can be considered: minimizing the number of substructures used as a screen, and optimizing a set of substructures under the limit of the number of substructures. The former corresponds to OPTIMAL SCREEN, and the latter corresponds to OPTIMAL SCREEN-II, which are to be defined soon. Although substructure screens have been applied to chemical structures so far, it may be applied to other objects for which searching for similar structures is important. For example, it may be applied to 3D protein structures, and mechanical/electrical CAD data.

Here, we define problems formally. Let $B = \{b_1, \cdots, b_n\}$ be a set of bit vectors, where each b_i consists of m bits. For b_i and an integer j, $b_i|j$ denotes j-th bit value of b_i. For a set of integers (bits) $bs = \{j_1, \cdots, j_k\}$, $b_i|bs$ denotes a projection of b_i to bs. For example, $1011011|\{1,2,3,4\} = 1011$ and $1011011|\{2,5,6,7\} = 0011$. Let $width(B, bs) = \max_{b_i \in B} |\{b_j : b_j|bs = b_i|bs\}|$. Then we define optimal screen problems as follows:

> **OPTIMAL SCREEN:** Given a set of bit vectors $B = \{b_1, \cdots, b_n\}$, find a minimum cardinality set of bits bs such that $width(B, bs) = 1$.
> **OPTIMAL SCREEN-II:** Given a set of bit vectors $B = \{b_1, \cdots, b_n\}$ and an integer L, find a set of bits bs of cardinality at most L which minimizes $width(B, bs)$.

The above problems are closely related to the *minimum cardinality key* problem in database theory [3, 12]. In this paper, we consider the following two versions of this problem:

> **MINIMUM KEY:** Given a set of functional dependencies $F = \{F_1, \cdots, F_n\}$ over a set of attributes $A = \{A_1, \cdots, A_m\}$, find a minimum cardinality key $K \subseteq A$ implied by F.
> **MINIMUM KEY-II:** Given a set of tuples $r = \{t_1, \cdots, t_n\}$ over a set of attributes $A = \{A_1, \cdots, A_m\}$, find a minimum cardinality key $K \subseteq A$ (i.e., find a minimum $K \subseteq A$ such that $(\forall i \neq j)(t_i|K \neq t_j|K)$ where $t_i|K$ denotes a projection of t_i to K).

Moreover we consider the following variants of the SET COVER problem, which are closely related to the above problems:

> **SET COVER-II:** Given a set $S = \{s_1, \cdots, s_m\}$ over U where $|U| = n$ and an integer L, find a subset $C \subseteq S$ of cardinality at most L which maximizes the number of covered elements in U.
> **SET COVER-III:** Given S and L, find a subset $C \subseteq S$ of cardinality at most L which minimizes the number of uncovered elements in U.

Recall that SET COVER is defined as below:

> **SET COVER:** Given a set S over U, find a minimum cardinality set (*cover*) $C \subseteq S$ such that $\bigcup_{s_i \in C} s_i = U$.

Since all the above problems are NP-hard, we consider *polynomial time approximation algorithms*. In particular, we consider upper and lower bounds of performance ratios. Recall that the *performance ratio* of an approximation algorithm for a maximization problem is the worst-case ratio of the size of the optimal solution to the size of the approximate solution (for a minimization problem, we use the inverse). Moreover we consider the *average case performance ratios* for optimal screen problems, where we assume that each instance of the same problem size appears with the same probability. Results are summerized in Table 1, where we assume $P \neq NP$ and $NP \not\subseteq DTIME(n^{polylog(n)})$, and ϵ is any constant such that $0 < \epsilon < 1$.

Table 1: Summary of results.

Problem	Upper bound	Lower bound
OPTIMAL SCREEN	$2 \ln n + 1$	$\Omega(\log n)$
(Average case)	$2 + \epsilon$	
OPTIMAL SCREEN-II		n^{ϵ}
(Average case)	$O(\log^{1+\epsilon} n)$	
MINIMUM KEY	?	$\Omega(\log n)$
MINIMUM KEY-II	$2 \ln n + 1$	$\Omega(\log n)$
SET COVER-II	$e/(e-1)$	MAX SNP-hard
SET COVER-III		n^{ϵ}

Here, we briefly review related work. A lot of studies have been done for approximation algorithms [2, 4, 8, 14]. SET COVER is one of the most important problems. Johnson and Lovász independently showed that SET COVER can be approximated within a factor of $\ln n + 1$ using a simple greedy algorithm [8, 11], from which several improvements followed [6]. Lund and Yannakakis proved that SET COVER can not be approximated within a factor of $\frac{1}{4} \log n$ under the assumption of $NP \not\subseteq DTIME(n^{polylog(n)})$. MINIMUM KEY and MINIMUM KEY-II were shown to be NP-complete [3, 12]. We found that SET COVER-III and OPTIMAL SCREEN-II are closely related to MIN 3NON-TAUTOLOGY, which is the problem of minimizing the number of satisfiable disjunctions, given a boolean formula in disjunctive normal form with three literals per disjunct [9]. Although Kolaitis and Thakur showed that MIN 3NON-TAUTOLOGY can not be approximated within any constant factor, their proof can be modified to showing n^{ϵ} lower bound for any constant $0 < \epsilon < 1$. We use similar proofs to obtain lower bounds for SET COVER-III and OPTIMAL SCREEN-II. Our average case analysis for optimal screen problems uses *Kolmogorov complexity*. Jiang and Li have already applied Kolmogorov complexity to the analysis of the average case performance ratio [7]. Our analysis is similar to their analysis.

2 Set cover

Hereafter we describe the proofs for the obtained results where details are omitted. In this section we consider variants of SET COVER. First we show that SET COVER-II is MAX SNP-hard, from which a lower bound of the constant size performance ratio follows [2, 14].

Theorem 2.1 *SET COVER-II is MAX SNP-hard.*
Proof. We prove it by means of an L-Reduction from MAX 2SAT-B [14]. From an instance of MAX 2SAT-B which consists of a set of clauses $C = \{c_1, \cdots, c_m\}$ over a set of of variables $X = \{x_1, \cdots, x_n\}$ where $c_i = \{c_i^1, c_i^2\}$ and c_i^j is a literal (x_k or $\overline{x_k}$), we construct an instance of SET COVER-II in the following way:

$$U = \{c_1, \cdots, c_m\} \cup \{d_i^j : 1 \le i \le m, 1 \le j \le 2B\}, \quad S = \{s_1, \overline{s_1}, s_2, \overline{s_2}, \cdots, s_n, \overline{s_n}\},$$
$$s_i = \{c_j : x_i \in c_j\} \cup \{d_i^1, \cdots, d_i^{2B}\}, \ \overline{s_i} = \{c_j : \overline{x_i} \in c_j\} \cup \{d_i^1, \cdots, d_i^{2B}\}, \quad L = n.$$

It is easy to see that the above reduction is an L-reduction. □

Next we show that SET COVER-II can be approximated within a factor of $e/(e-1)$ using a well-known greedy algorithm [8, 11].

Theorem 2.2 *SET COVER-II can be approximated within a factor of $e/(e-1)$ using a simple greedy algorithm.*
Proof. Given a family of sets $S = \{s_1, s_2, \cdots, s_n\}$ and an integer L, the greedy algorithm for SET COVER-II is as follows:

```
for $i$:=1 to $n$ do $s_i^1$:=$s_i$;
for $j$:=1 to $L$ do begin
  pick $s_k^j$ such that $|s_k^j| \ge |s_i^j|$ for any $i$;  output $s^j$:=$s_k^j$;
  for $i$:=1 to $n$ do $s_i^{j+1}$:=$s_i^j - s_k^j$ end.
```

The solution of the greedy algorithm is $\cup_{j=1}^L s^j$. Next we show that $|\cup_{j=1}^L s_{i_j}|/|\cup_{j=1}^L s^j| \le \frac{e}{e-1}$ for any $1 \le i_1, i_2, \cdots, i_L \le n$, where e is the nature number. From the greedy algorithm, it is not difficult to see the following facts:

$$L|s^1| \ge |\cup_{j=1}^L s_{i_j}|, \quad |s^1| + L|s^2| \ge |\cup_{j=1}^L s_{i_j}|, \quad \cdots, \quad |s^1| + \cdots + |s^{L-1}| + L|s^L| \ge |\cup_{j=1}^L s_{i_j}|.$$

Hence, we transform the upper bound problem of SET COVER-II into a linear programming problem: finding the maximum value of $\min\{\frac{Lx_1}{x_1+x_2+\cdots+x_L}, \frac{x_1+Lx_2}{x_1+x_2+\cdots+x_L}, \cdots, \frac{x_1+\cdots+x_{L-1}+Lx_L}{x_1+x_2+\cdots+x_L}\}$. We will prove later that it reaches maximum when $Lx_1 = x_1+Lx_2 = \cdots = x_1+\cdots+x_{L-1}+Lx_L$. From $Lx_1 = x_1+Lx_2 = \cdots = x_1+\cdots+x_{L-1}+Lx_L$, we have $x_i = (1-\frac{1}{L})^{i-1}x_1$ for $i = 2, 3, \cdots, L$. Hence, we obtain the conclusion that the maximum value is equal to $\frac{1}{1-(1-\frac{1}{L})^L} < \frac{e}{e-1}$. Moreover we can prove that this bound is tight where we omit the proof here.

Now we show that $m = \min\{\frac{Lx_1}{x_1+x_2+\cdots+x_L}, \frac{x_1+Lx_2}{x_1+x_2+\cdots+x_L}, \cdots, \frac{x_1+\cdots+x_{L-1}+Lx_L}{x_1+x_2+\cdots+x_L}\}$ reaches maximum when $Lx_1 = x_1 + Lx_2 = \cdots = x_1 + \cdots + x_{L-1} + Lx_L$. Here we let

$$\frac{Lx_1}{x_1+x_2+\cdots+x_L} = m+t_1, \quad \frac{x_1+Lx_2}{x_1+x_2+\cdots+x_L} = m+t_2, \quad \cdots, \quad \frac{x_1+\cdots+x_{L-1}+Lx_L}{x_1+x_2+\cdots+x_L} = m+t_L$$

where $t_1, \cdots, t_L \ge 0$. On the other hand, there exist positive constants $c_1, \cdots, c_L$ such that

$$c_1(Lx_1) + c_2(x_1 + Lx_2) + \cdots + c_L(x_1 + \cdots + x_{L-1} + Lx_L) \ = \ L(x_1 + x_2 + \cdots + x_L)\ .$$

Hence, we have $m = \frac{L-\sum_{i=1}^L c_i t_i}{\sum_{i=1}^L c_i}$. So m reaches maximum when $t_1 = t_2 = \cdots = t_L = 0$. □

Note that if we let $L = n$ and repeat the greedy algorithm $\lceil \ln n \rceil + 1$ time, the number of uncovered elements becomes 0 since $n(1 - \frac{e-1}{e})^{\lceil \ln n \rceil + 1} < 1$. It matches $\ln n + 1$ performance ratio of the greedy algorithm for original SET COVER [8, 11]. Thus we can see that the ratio of the number of uncovered elements steadly decreases by the greedy algorithm.

Next we consider SET COVER-III. Since the performance ratio can become ∞ if the number of uncovered elements in an optimal solution is 0, we consider the case where the number of uncovered elements in an optimal solution is at least 1.

Theorem 2.3 *SET COVER-III can not be approximated within a factor of n^ϵ in polynomial time for any constant $0 < \epsilon < 1$ unless $P = NP$.*

Proof. We use a similar technique as in [9] although an additional technique is introduced.

Let $S' = \{s_1, \cdots, s_M\}$ over $U' = \{x_1, \cdots, x_N\}$ be an instance of SET COVER. We consider the problem of deciding whether or not the size of a cover is at most K. From this instance, we construct an instance S of SET COVER-III making H copies of 3-SET COVER in the following way where $H = N^k$ for some integer $k > 0$:

$$S = \{s_i^j : 1 \le i \le M, 1 \le j \le H\}, \ U = \{x_i^j : 1 \le i \le N, 1 \le j \le H\} \cup \{x_0\} \text{ and}$$
$$L = HK, \text{ where } x_0 \text{ is a new element not appeared in } U' \text{ and } s_i^j = \{x_k^j : x_k \in s_i\}.$$

Then it is easy to see that the following claim holds.

Claim. *If S' does not have a cover of size K, the number of uncovered elements in U is at least $H + 1$.*

If we let $opt(S, L)$ be the minimum number of uncovered elements in SET COVER-III, the following relation holds: $opt(S, L) = 1$ if S' has a cover of size K, $opt(S, L) > H$ otherwise. Therefore, if SET COVER-III can be approximated within a factor of H, then SET COVER can be solved. Here note that $n = N^{k+1} + 1$ and $H = (n-1)^{\frac{k}{k+1}}$ since $H = N^k$. Given any $0 < \epsilon < 1$, $(n-1)^{\frac{k}{k+1}} > n^\epsilon$ can hold for sufficiently large n by choosing an appropriate integer $k > 0$. Thus the theorem is proved. □

Note that, using a similar discussion, we can show that MIN 3NON-TAUTOLOGY can not be approximated within a factor of n^ϵ for any $0 < \epsilon < 1$ unless $P = NP$ [9]. This result suggests that the same lower bound holds for many minimization problems [9].

3 Minimum cardinality key

Two problems are considered for finding minimum cardinality keys. In one problem, an input is given in the form of a set of functional dependencies. In the other problem, an input is given in the form of a set of tuples. First we consider the former problem.

Theorem 3.1 *MINIMUM KEY can not be approximated within a factor of $c \log n$ in polynomial time for some constant c unless $NP \subseteq DTIME(n^{polylog(n)})$.*

Proof. We show an approximation preserving reduction from SET COVER. Let $S = \{s_1, \cdots, s_M\}$ over $U = \{1, \cdots, N\}$ be an instance of SET COVER, where we assume $M = O(poly(N))$ holds. $\frac{1}{4} \log N$ lower bound was derived for such a case [13].

From this instance, we construct an instance of MINIMUM KEY in the following way:

$$A = \{A_1, \cdots, A_M\} \cup \{A_{i,j} \mid 1 \le j \le N, 1 \le h \le H\},$$
$$F = \{A_i \to A_{j,h} \mid j \in s_i, 1 \le h \le H\} \cup \{A_{1,1} \cdots A_{n,H} \to A_i \mid 1 \le i \le M\},$$

where H is an integer satisfying $H = c'M$ for some large fixed constant c'. Then the following property holds: a key K must contain at least H elements of F if $K \cap \{A_1, \cdots, A_M\}$ is not a key. Thus each key K such that $|K| < H$ corresponds to a set cover. Since $\log n = \log |A| = \log(M + c'MN) = O(\log N)$, the theorem holds. □

Next we consider MINIMUM KEY-II in which an input is given in the form of a set of tuples.

Theorem 3.2 *MINIMUM KEY-II can not be approximated within a factor of* $\frac{1}{4}\log n$ *in polynomial time unless* $NP \subseteq DTIME(n^{polylog(n)})$.
Proof. As in Thm. 3.1, we prove it by means of an approximation preserving reduction from SET COVER. From an instance $S = \{s_1, \cdots, s_m\}$ over $U = \{1, \cdots, n\}$ of SET COVER, we construct a set of tuples $r = \{t_1, \cdots, t_n\}$ over $A = \{A_1, \cdots, A_m\}$ in the following way:

$$t_i|A_j = \begin{cases} i, & i \in s_j, \\ 0, & \text{otherwise.} \end{cases}$$

In this case, a key corresponds to a set cover in a one-to-one way and the theorem follows. □

Theorem 3.3 *MINIMUM KEY-II can be approximated within a factor of* $2\ln n + 1$ *in polynomial time.*
Proof. We reduce MINIMUM KEY-II to SET COVER. Let $r = \{t_1, \cdots, t_n\}$ over $A = \{A_1, \cdots, A_m\}$ be an instance of MINIMUM KEY-II. Remember that K is a key if and only if $(\forall i \neq j)(\exists A_k \in K)(t_i|A_k \neq t_j|A_k)$ holds. Then we construct an instance of SET COVER in the following way:

$$S = \{s_1, \cdots, s_m\},\ U = \{(t_i, t_j) : i < j\},\ \ s_k = \{(t_i, t_j) : t_i|A_k \neq t_j|A_k\}.$$

In this case, a set cover corresponds to a key in a one-to-one way. Since $\ln |U| + 1 \leq \ln n^2 + 1 = 2\ln n + 1$, the theorem holds. □

4 Optimal substructure screen

In this section we consider optimal substructure screen problems. First we consider the worst-case performance ratios as usual. Since a very high lower bound is proved for OPTIMAL SCREEN-II, we next consider the average-case performance ratios. The average-case analysis is important since it seems that the worst case seldom occurs in a practical case.

4.1 Worst case performance ratio

Note that OPTIMAL SCREEN can be considered as a special case of MINIMUM KEY-II where a value of each item is restricted to 0 or 1. Thus the following corollary immediately holds.

Corollary 4.1 *OPTIMAL SCREEN can be approximated within a factor of* $2\ln n + 1$ *in polynomial time.*

Although a slight modification (omitted here) is required, the following hardness result holds as in Thm. 3.2.

Theorem 4.2 *OPTIMAL SCREEN can not be approximated within a factor of* $c \log n$ *in polynomial time for some constant* c *unless* $NP \subseteq DTIME(n^{polylog(n)})$.

For OPTIMAL SCREEN-II, a very strong lower bound can be proved as in Thm. 2.3.

Theorem 4.3 *OPTIMAL SCREEN-II can not be approximated within a factor of* n^{ϵ} *in polynomial time for any constant* $0 < \epsilon < 1$ *unless* $P = NP$.

4.2 Average case performance ratio

Reference [7] is the first paper to analyze the average performance ratio of approximation problem with Kolmogorov complexity. Here we give a similar average case analysis on both OPTIMAL SCREEN and OPTIMAL SCREEN-II. We show that the former can be approximated within $2+\epsilon$ on average and the latter can be approximated within $\log^{1+\epsilon} n$ on average.

Let Σ be the alphabet $\{0,1\}$. Fix a universal Turing machine U with input and output alphabet Σ. The Kolmogorov complexity of a string $x \in \Sigma^*$ is defined as the length of the minimum program of U which outputs x, i.e., $K(x) = min\{|p| : U(p) = x\}$. Let $f(n)$ be a positive function. For any binary string x of length n, x is called Kolmogorov f-random if $K(x) \geq n - f(n)$. The number of Kolmogorov f-random strings in Σ^n is at least $2^n - 2^{n-f(n)} + 1$ [10].

For convenience, we suppose $m \leq n$. Our results are valid for the case $m = O(poly(n))$.

Theorem 4.4 *For any constant $\epsilon > 0$, OPTIMAL SCREEN can be approximated within $2+\epsilon$ on average.*

Proof. Consider an instance of OPTIMAL SCREEN, a set of binary vectors $B = \{b_1, b_2, \cdots, b_n\}$, where each b_i consists of m bits. We also treat B as a binary string of length nm, $b_1 b_2 \cdots b_n$. It can be distinguished whether B is a set of vectors or a binary string from the context. Let bs be a subset of $\{1, 2, \cdots, m\}$. Define $dup(B, bs) = |\{i :$ there exist $j < i$ such that $b_i|bs = b_j|bs\}|$. Apparently, $width(B, bs) = 1$ if and only if $dup(B, bs) = 0$. In the following, we use $bs(k)$ to denote the first k bit positions, i.e., $bs(k) = \{1, 2, \cdots, k\}$. And $\log n$ denotes the length of the binary form of n.

Lemma 4.5 *Let $c > 0$ be any constant. For sufficiently large n, if $dup(B, bs((2+c)\log n)) > \frac{4}{c}$, then we have Kolmogorov complexity $K(B) < nm - \log n$.*

Proof. Let $L = (2+c)\log n$ and $k = \frac{4}{c}$. Suppose $dup(B, bs(L)) > k$. There exist $j_1 < i_1, j_2 < i_2, ..., j_k < i_k$ such that $b_{j_1}|bs(L) = b_{i_1}|bs(L)$, $b_{j_2}|bs(L) = b_{i_2}|bs(L)$, $\cdots$, $b_{j_k}|bs(L) = b_{i_k}|bs(L)$. We can generate B by duplicating $b_{j_h}|bs(L)$ to $b_{i_h}|bs(L)$, $h = 1, \cdots, k$, provided that we are given $k, j_1, i_1, \cdots, j_k, i_k, L, n, m$ and the binary string obtained by removing $b_{i_1}|bs(L), b_{i_2}|bs(L), \cdots, b_{i_k}|bs(L)$ from B. (we denote this string by $B(L; -i_1, \cdots, -i_k)$)

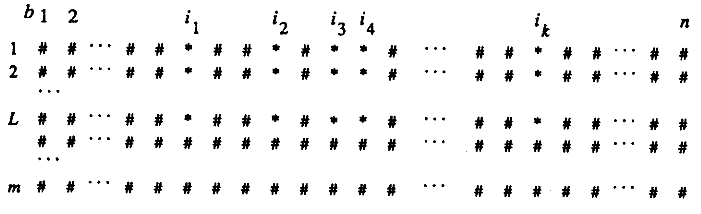

Figure 2: $B(L; -i_1, \cdots, -i_k)$ is the substring of B composed of #.

Encoding $k, j_1, i_1, \cdots, j_k, i_k, L, n, m$ in self-delimiting version, we can prove $K(B) \leq nm - kc\log n + 2\log n + O(k \log\log n)$. Since $k = \frac{4}{c}$, we have $K(B) < nm - \log n$ for sufficiently large n.

In this proof we imply that both $\frac{4}{c}$ and $(2+c)\log n$ are integers. The proof is valid by taking their integer parts if they are not integers. □

Consider the following naive algorithm: *c-algorithm*

Let $bs = bs((2+c)\log n)$. If $b_i|bs = b_j|bs$ for some $i \neq j$, add a bit position h ($h \in \{1, 2, \cdots, m\}$) to bs (i.e. $bs{:=}bs \cup \{h\}$) such that $b_i|bs \neq b_j|bs$. Repeat until $(\forall i \neq j)(b_i|bs \neq b_j|bs)$.

For any instance B, denote the bs obtained by applying *c-algorithm* to B by *c-algorithm*(B). From Lemma 4.5, we have $|\{B : |c\text{-}algorithm(B)| > (2+c)\log n + \frac{4}{c}\}| \leq \frac{2^{nm}}{n}$. Since b_i must be different from the other b_j's, it is apparent that $|opt(B)| \geq \log n$ and $|c\text{-}algorithm(B)| \leq m$ for any instance B. The number of all possible instances is $2^m(2^m-1)\cdots(2^m-n+1)$. Since $m > 2\log n$(otherwise the approximation ratio can never exceed 2), we have $2^m - n > (1-\frac{1}{n})2^m$, $2^m(2^m-1)\cdots(2^m-n+1) > ((1-\frac{1}{n})2^m)^n > \frac{1}{4}2^{nm}$. Hence, the average approximation ratio of OPTIMAL SCREEN is less than

$$\frac{(\frac{1}{4}2^{nm} - 2^{nm}/n)((2+c)\log n + 4/c) + (2^{nm}/n)m}{\frac{1}{4}2^{nm}\log n} < 2 + c + \frac{1}{\log n}(4\frac{m}{n} + \frac{16}{c})$$

It is less than $2 + 2c$ for sufficiently large n $(m \leq n)$. □

Using a similar technique, we can prove the following theorem, where we omit the proof.

Theorem 4.6 *For any constant $\epsilon > 0$, OPTIMAL SCREEN-II can be approximated within $\log^{1+\epsilon} n$ on average.*

Acknowledgement

Tatsuya Akutsu is partially supported by the Grant-in-Aid for Scientific Research on Priority Areas, "Genome Informatics", of the Ministry of Education, Science and Culture of Japan.

References

[1] S. Anderson. Graphical representation of molecules and substructure-search queries in MACCS. *J. Molecular Graphics*, 2:83–89, 1984.

[2] S. Arora, C. Lund, R. Motwani, M. Sudan and M. Szegedy. Proof verification and hardness of approximation algorithms. *Proc. FOCS'92*, 14–23.

[3] C. Beeri, M. Dowd, R. Fagin and R. Statman. On the structure of armstrong relations for functional dependencies. *J. ACM*, 31:30–46, 1984.

[4] P. Crescenzi and V. Kann. *A compendium of NP optimization problems*, Manuscript, 1995.

[5] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*, Freeman, San Francisco, 1979.

[6] M. M. Halldórsson. Approximating discrete collections via local improvement. *Proc. SODA '95.*

[7] T. Jiang and M. Li. On the approximation of shortest common supersequences and longest common subsequences. *Proc. ICALP'94*, LNCS, Springer-Verlag, 191–202.

[8] D. S. Johnson. Approximation algorithms for combinatorial problems. *JCSS*, 9:256–278, 1974.

[9] P. G. Kolaitis and M. N. Thakur. Logical definability of NP optimization problems. *Information and Computation*, 115:321-353, 1994.

[10] M. Li and P.M.B. Vitanyi. Kolmogorov complexity and its application. *Handbook of Theoretical Computer Science*, Vol. A, 187-254, Elsevier/MIT Press, 1990.

[11] L. Lovász. On the ratio of optimal integral and fractional covers. *Disc. Math.*, 13:383–390, 1975.

[12] C. L. Lucchesi and S. L. Osborn. Candidate keys for relations. *JCSS*, 17:270–279, 1978.

[13] C. Lund and M. Yannakakis. On the hardness of approximating minimization problems. *J. ACM*, 41:960–981, 1994.

[14] C. H. Papadimitriou and M. Yannakakis. Optimization, approximation, and complexity classes. *JCSS*, 43:425–440, 1991.

[15] R. E. Stobaugh. Chemical substructure searching. *J. Chemical Information and Computer Sciences*, 25:271–275, 1985.