数理解析研究所講究録
第381巻 1980年 73-84

プログラム図式における Time と Space の Trade-off

笠井琢美, 足立暁生

§1 序

Complexity Theory の中の重要な問題に time と space の trade-off 問題と呼ばれる未解決の問題がある。time と space の trade-off 問題とは, ある問題が存在して, その問題を解くのに制限された space では多くの時間を必要とするが, 十分 space が使えれば time を節約できる, といった問題が実際に存在するかといった問題である. ここではこの問題をプログラム図式の上で考えてみる. そしてソーティングなどのいくつかの図式において, time と space の trade-off が実際に起ることを証明する.

プログラム図式とはその中に現われる関数や述語を特定のものに固定しないものをいう. たとえばソーティングを実行する図式では, 比較演算子 $\leq$ は任意の順序集合 D 上の関係で, 順序の公理を満せばなんであってもよい.

プログラム図式を実行する媒体として, 本論文では "relativized Turing machine" を考える. relativized Turing machine は1本の読み取り専用の入力テープ, 1本の一方向にしか動かない書き込み専用の出力テープと数本の補助テープ

を持っている．さらに有限個の register を持っている．Dを与えられた domain とするとき，各テープのます目と register に Dの元を書き込むことができる．ある register の内容をあるテープの head が示すます目に書き込むこともできるし，逆にあるテープの head の示すます目の内容を register に持ってくることもできる．今，X_i, X_{j_1}, $\cdots$, X_{j_m} を register，f を m 変数関数とするとき

$$X_i \leftarrow f(X_{j_1}, \cdots, X_{j_m})$$

の形の命令が許され，また p を m 変数述語とするとき

$$p(X_{j_1}, \cdots, X_{j_m})$$

の形の判定が許される．ただし入力テープと出力テープのます目は数えない．このようなモデルの上で次が成立する．

a を $0 \leq a < 1$ なる数とする．このとき n 個の元をソートする $O(n^a)$ space bounded が，$O(n^{2-a}\log n)$ time bounded な回式が存在する．さらに $O(n^a)$ space bounded などんな回式も $O(n^{2-a})$ time 以上の時間を必要とする．したがってソーティングに関しては time と space の trade-off が実際に起る．ここでは，time complexity の上界 $O(n^{2-a}\log n)$ と下界 $O(n^{2-a})$ には少し差がある．しかし入力テープへの書き込みを許した場合には次が成立する．ソーティングを実行する $S(n^a)$ space bounded な書き込み可能入力

テープを持つ Turing machine では $O(n^{2-a})$ time が必要かつ十分である. ただし入力テープへの書き込みを許した場合でも, いくら入力テープのます目を使用しても space complexity には数えないこととする.

さらに次のような問題 Q が存在することが示される. Q は $O(n^2)$ space, polynomial time で計算できる. また Q は $O(n \log n)$ space, exponential time で計算できる. さらに Q を解くどんな $S(n)$ space bounded な図式も, $S(n) \ll n^2$ なら exponential time 以上の時間を必要とする.

§2 基本モデル

ここではプログラム図式(以後単に図式と呼ぶ), および図式を実行する媒体である計算機モデルについて述べる. いままでにすでにいくつかの図式のクラスが導入されている. たとえば Ianov shema, recursive schema, array を持つ図式, 自然数を扱うことのできる図式, pushdown stack を持つ図式などがある. 本論文で扱う図式は, その中で最も強力なもの, すなわち配列を持つ図式と同じ表現能力を持つ.

定義 2.1. 関数記号と述語記号の有限列 $\tau = (f_1, f_2, \cdots, f_k; P_1, P_2, \cdots, P_m)$ を 型 と呼ぶ. 各々の関数記号と述語記号に

は自然数が対応づけられている。f_i に n が対応づけられているとき、f_i を n変数関数記号、p_i に n が対応づけられているとき p_i を n変数述語記号 と呼ぶ。0変数関数記号を定数記号と呼ぶ。以後すべての型は定数記号 Λ と $\#$ 1変数述語記号 blank と mark とを含むものとする。したがって型 $(f_1, \cdots, f_k ; p_1, \cdots, p_m)$ と書いたときは $(\Lambda, \#, f_1, f_2, \cdots, f_k ; \text{blank}, \text{mark}, p_1, \cdots, p_m)$ の略であるとする。

$\tau = (f_1, \cdots, f_k ; p_1, \cdots, p_m)$ を型とする。このとき型 τ の解釈とは $\mathcal{I} = (D ; \bar{f}_1, \cdots, \bar{f}_k ; \bar{p}_1, \cdots, \bar{p}_m)$ のことをいう。ここで

(i) D は空でない集合。$\bar{\Lambda}$ と $\bar{\#}$ を D の元でない抽象的な元とし、$\hat{D} = D \cup \{\bar{\Lambda}, \bar{\#}\}$ とおく。(ii) f_i が n 変数関数記号なら $\bar{f}_i$ は D^n から D への関数。特に $n=0$ (即ち f_i が定数記号) なら $\bar{f}_i$ は D の元である。$\bar{f}_i$ は次のように $\hat{D}^n$ から $\hat{D}$ への関数に拡張できる。

$$\bar{f}_i(d_1, \cdots, d_n) = \bar{\Lambda} \quad \text{if } d_i = \bar{\Lambda} \text{ or } d_i = \bar{\#} \text{ for some } i.$$

(iii) p_i が n 変数述語記号なら $\bar{p}_i$ は D^n 上の述語 (即ち $\bar{p}_i$ は D^n の部分集合)。また $\overline{\text{blank}} = \{\bar{\Lambda}\}$, $\overline{\text{mark}} = \{\bar{\#}\}$ と定める。(即ち $\overline{\text{blank}}(d)$ が成立 $\Leftrightarrow d = \bar{\Lambda}$, $\overline{\text{mark}}(d)$ が成立 $\Leftrightarrow d = \bar{\#}$ である)

定義 2.2. relativised Turing machine (以後単に TM と書く) は1本の2方向読み取り専用の入力テープと, 1本の1方向書き込み専用の出力テープと, 有限個の2方向読み書き可能な補助テープを持っている. 各テープは右方向に無限とする. さらに有限個の register を持っている. さて TM の機械語を定義しよう. 機械語はテープ名

$$T_0, T_1, T_2, \cdots$$

と register 名

$$X_0, X_1, X_2, \cdots$$

を使用する. T_0 は入力テープの名前, T_1 は出力テープの名前である. $\tau = (f_1, \cdots, f_k : p_1, \cdots, p_m)$ を型とする. このとき 型 τ の TM の命令とは次のようなる.

(i) LOAD X_j, T_i (ii) STORE X_j, T_i

(iii) MVR T_i (iv) MVL T_i

(v) $X_i \leftarrow f_j(X_{k_1}, \cdots, X_{k_m})$ (vi) IF $p_j(X_{k_1}, \cdots, X_{k_m})$ GOTO l

ここで, $k_1, \cdots, k_m, l$ は自然数であり, f_j は n 変数関数記号, p_j は n 変数述語記号である. (i) と (iv) では $i \neq 1$, (ii) では $i \neq 0$ でなければならない.

命令の有限列

$$P = A_0 A_1 \cdots A_t$$

を型 τ のプログラム図式（あるいは単に図式）と呼ぶ．

$\mathcal{I} = (D; \bar{f}_1, \cdots, \bar{f}_k; \bar{P}_1, \cdots, \bar{P}_m)$ を型 τ の解釈とする．D^* の元 $x = d_1 d_2 \cdots d_n$ が入力として与えられたとしよう．このとき入力 x に対する P の計算は次のように行なわれる．P は次のような状態（初期 configuration）から計算を開始する．入力テープには #$d_1 d_2 \cdots d_n$# が書かれている．即ち 0 番目のます目に #，i 番目のます目に d_i が $(1 \leq i \leq n)$，$n+1$ 番目のます目に # が書かれており，その他のテープのます目および register には Λ (blank) が書かれている．また各テープの head は最左端にある．実行は P の最初の命令 A_0 から順になされる．STORE X_j, T_i は register X_j の内容をテープ T_i の head が示すます目に書き込む，LOAD X_j, T_i はテープ T_i の head が示すます目の内容を register X_j に書き込む，MVR T_i はテープ T_i の head を右に一こま移動する，MVL T_i はテープ T_i の head を左に一こま移動する，$X_i \leftarrow f_j(X_{k_1}, \cdots, X_{k_m})$ は $X_{k_1}, \cdots, X_{k_m}$ の内容を $d_1, \cdots, d_m$ とするとき値 $f_j(d_1, \cdots, d_m)$ を X_i に書き込む，IF $P_j(X_{k_1}, \cdots, X_{k_m})$ GOTO l は $X_{k_1}, \cdots, X_{k_m}$ の内容を $d_1, \cdots, d_m$ とするとき $P_j(d_1, \cdots, d_m)$ の値が真なら，次に l 番目の命令を実行する，もしそうでないなら，現在実行中の命令の直後の命令を実行することを意味する．次に実行する命令がないなら，実行は終了する．

計算が終了したときの出力テープの内容を $e_1 e_2 \cdots e_n \Lambda\Lambda\cdots$, $e_i \in \hat{D}$, $1 \le i \le n$, $e_n \neq \Lambda$ とする. このとき $y = e_1 \cdots e_n$ $(n \ge 0)$ が P の入力 x に対する出力であり, $\bar{P}_\sigma(x) = y$ と書く. このように定義される部分関数 $\bar{P}_\sigma : D^* \to \hat{D}^*$ を σ のもとで P によって実現される部分関数という. time complexity は計算中に実行された命令の個数で, これを $time_{P,\sigma}(x)$ で示す. space complexity は計算中に使用された register の個数と使用された補助テープのます目の個数の和で, これを $space_{P,\sigma}(x)$ で示す. T と S を N から N への関数とする. (ここで N は自然数全体からなる集合). ただし, 任意の n に対し $T(n) \ge n$ とする. このとき P が σ のもとで T-time bounded (S-space bounded) であるとは, 任意の入力 $x \in D^*$ に対し

$$time_{P,\sigma}(x) \le T(|x|)$$

$$(\ Space_{P,\sigma}(x) \le S(|x|)\ )$$

が成立するときをいう. $\mathbb{C}$ を型の解釈のクラスとする. このとき図式 P がクラス $\mathbb{C}$ のもとで T-time bounded (S-space bounded) であるとは $\mathbb{C}$ の任意の元 σ に対し, P が σ のもとで T-time bounded (S-space bounded) であるときをいう.

定義2.3　上で定義したTMは，入力テープへの書き込みを禁止した．しかし入力テープも補助記憶として使用したいこともたびたびある．入力テープへの書き込みが許されるような機械を，書き込み可能な入力テープを持つTMと呼ぶ．また書き込み可能入力テープを持つTMでは，入力テープが出力テープの代用をすることがある．即ち計算が終ったとき入力テープに書かれている内容がこの機械の出力である．
特にn-テープ Turing machine とは1本の読み書き可能な入出力テープと，n-1本の読み書き可能な補助テープを持つ機械のことをいう．

P_1とP_2が同じ型での図式とする．$\mathcal{I}$を型での解釈とする．このときP_1とP_2が$\mathcal{I}$のもとで同値であるとは，任意の入力に対し，P_1とP_2が$\mathcal{I}$のもとで共に止まないか，または共に停止し同じ出力を持つときをいう．

relativised TMの場合も，通常のTMと同様な次が成立する．

定理2.1　任意の図式P_1に対し，1テープTM上の図式P_2が存在し，任意の解釈$\mathcal{I}$に対し次が成立する．

(i)　P_1とP_2は$\mathcal{I}$のもとで同値

(ii)　P_1が$\mathcal{I}$のもとで$T(n)$ time boundedなら，P_2は$\mathcal{I}$のもとで$O(T(n)^2)$ time bounded

証明　Hartmanis and Stearns [　] と同様

定理 2.2.　任意の図式 P_1 に対し，2テープ TM 上の図式 P_2 が存在し，任意の解釈 I に対し次が成立する．

(i)　P_1 と P_2 は I のもとで同値

(ii)　P_1 が I のもとで $T(n)$ time bounded なら P_2 は I のもとで $O(T(n) \log T(n))$ time bounded.

証明　Hennie and Stearns [　] と同様．

以下でいくつかの問題に対する図式を構成する．この構成においては，TM の機械語でなく高級言語（Pidgin ALGOL）で書く．このとき TM の各テープは一次元配列 表わす．また head の位置は整数型の変数で表わす．たとえば i をあるテープの head を表わす変数とするとき，$i \leftarrow 0$ は head i を最左端に動かすことを意味する．（したがって $i \leftarrow 0$ の実行に必要な時間は，その時の head の位置に依存する．

以後関数記号とある解釈のもとでそれの表わす関数，および述語記号とそれの表わす述語とを混同して用いる．たとえば Λ は空白を表わす定数記号であると同時に空白という定数自身を表わす．

§3 得られた結果の概要

以下得られた主要な結果だけを述べる.（完全な論文もありますから，希望者は申し出下さい）

定理 sortingを実行する任意の $S(n)$ space $T(n)$ time bounded な回式に対し，次が成立する.

$$T(n) \geq \frac{n^2}{4S(n)}$$

入力テープへの書き込みを許しても同じ結果が成立する.
（入力テープへの書き込みを許した場合でも，入力テープへの書き込みは space complexity には数えないこととする.）

定理 S を $1 \leq S(n) \leq n/\log n$ なる関数とする. このとき $O(S(n))$ space, $O(n^2/S(n))$ time bounded な書き込み可能入力テープを持，TM上の sorting を実行する回式が存在する.

定理 $1 \leq S(n) \leq n/\log n$ とする. このとき sorting を実行する $O(S(n))$ space, $O(n^2 \log n / S(n))$ time bounded な回式が存在する.

したがって sorting に関しては次のような time と space の trade off が成立する

○書込み可能な入力テープを持つTM

space	time
$O(1)$	$O(n^2)$
$O(S(n))$	$O(\frac{n^2}{S(n)})$
$O(n)$	$O(n \log n)$

○入力テープへの書き込みを禁止したとき

space	time の下界	time の上界
$O(1)$	$O(n^2)$	$O(n^2)$
$O(S(n))$	$O(\frac{n^2}{S(n)})$	$O(\frac{n^2 \log n}{S(n)})$
$O(n)$	$O(n \log n)$	$O(n \log n)$

定理　次のような問題 Q が存在する

(i) Q は $O(n^2)$ space polynomial time で解ける

(ii) Q は $O(n \log n)$ space で解ける

(iii) Q を解くどんな $S(n)$ space bounded な回式も $S(n) \ll n^2$ なら 2^n 時間以上を必要とする.

References

[1] Aho, A. V., J. E. Hopcroft, and J. D. Ullman, The Design and Analysis of Computer Algorithms, Addison-Wesley, Reading, Mass., 1974

[2] Baker, T., J. Gill and R. Solovary, Relativization of P = ? NP question, SIAM J. Comput., 4 (1975) 431-442

[3] Hartmanis, J. and J. E. Hopcroft, Independence results in computer science, SIGACT News 13-24, 1976

[4] Hartmanis, J., and R. E. Stearns, On the computational complexity of algorithms, Trans. Amer. Math. Soc., 117 (1965), 285-306

[5] Hennie, F. C., and R. E. Stearns, Two tape simulation of multitape machines, J. ACM 13 (1966) 533-546.

[6] Kasai, T., A. Adachi and S. Iwata, Classes of Pebble Games and Complete Problems, SIAM J. Comput. Vol.8 pp. 574∿586 (1979)

[7] Kasai, T., and A. Adachi, A Characterization of Time Complexity by Simple Loop Programs, to appear in J. Computer and System Sciences

[8] Knuth, D. E., The art of computer programming, Vol.3, Addison-Wesley Reading Mass., (1973)

[9] Paul, W. J., and R. E. Tarjan, Time-space trade-offs in a pebble game, Technical Report, STAN-CS-77-619 (1977)

[10] Pippenger, N., A time-space trade-off, IBM Research Report RC 6550 (1977)