SIAM J. COMPUT.
Vol. 19, No. 4, pp. 673-677, August 1990

007

A NOTE ON BENNETT'S TIME–SPACE TRADEOFF FOR REVERSIBLE COMPUTATION*

ROBERT Y. LEVINE† AND ALAN T. SHERMAN‡

Abstract. Given any irreversible program with running time T and space complexity S, and given any $\varepsilon > 0$, Bennett shows how to construct an equivalent reversible program with running time $O(T^{1+\varepsilon})$ and space complexity $O(S \ln T)$. Although these loose upper bounds are formally correct, they are misleading due to a hidden constant factor in the space bound. It is shown that this constant factor is approximately $\varepsilon 2^{1/\varepsilon}$, which diverges exponentially as ε approaches 0. Bennett's analysis is simplified using recurrence equations and it is proven that the reversible program actually runs in time $\Theta(T^{1+\varepsilon}/S^{\varepsilon})$ and space $\Theta(S(1+\ln(T/S)))$.

Bennett claims that for any $\varepsilon > 0$, the reversible program can be made to run in time $O(T)$ and space $O(ST^{\varepsilon})$. This claim is corrected and tightened as follows: whenever $T \geqq 2S$ and for any $\varepsilon \geqq 1/(0.58 \lg(T/S))$, the reversible program can be made to run in time $\Theta(T)$ and space $\Omega(S(T/S)^{\varepsilon/2}) \cap O(S(T/S)^{\varepsilon})$. For $S \leqq T < 2S$, Bennett's 1973 simulation yields an equivalent reversible program that runs in time $\Theta(T)$ and space $\Theta(S)$.

Key words. algorithms, reversible computation, time-space tradeoff

AMS(MOS) subject classifications. 68Q05, 68Q15

1. Introduction.

A Turing machine is *reversible* if and only if its state-transition function is injective. In other words, a program is reversible if for any input and for any state in the program execution on that input, the preceding state is uniquely determined from the current state. For example, the program "On input x and y, output $x+y$." is not reversible because the input cannot be determined from the output, but the related program "On input x and y, output $(x+y, x)$." is reversible. The notion of reversible computation might someday radically alter the design of computers because there are models of reversible computation in which computations do not dissipate heat [3].

In 1973 Bennett [2] presented a general method for transforming any irreversible program into an equivalent reversible program. This method works by reversibly simulating the irreversible program. Let T and S denote, respectively, the time and space bounds of the irreversible program. Bennett proved that his simulation runs in time $\Theta(T)$ and space $\Theta(T+S)$. Thus, because T can be exponential in S, the simulation takes space exponential in S in the worst case.

In 1988 Bennett [1] improved his simulation by introducing a time-space tradeoff. He proved the following result: given any $\varepsilon > 0$, the revised simulation can be made to run in time $O(T^{1+\varepsilon})$ and space $O(S \ln T)$.[1] Although these loose upper bounds are formally correct, they are misleading because there is a large hidden constant factor in the space bound that depends on ε; in fact, the tradeoff is between the exponent in the reversible time and the constant factor in the reversible space. In this note we exactly compute the constant factor in the space bound and show that it is approximately

* Received by the editors May 15, 1989; accepted for publication (in revised form) October 5, 1989.

† Massachusetts Institute of Technology, Lincoln Laboratory, Lexington, Massachusetts 02173-0073. This work was done while this author was a student at Tufts University.

‡ Computer Science Department, University of Maryland, Baltimore, Maryland 21228, and Institute for Advanced Computer Studies, University of Maryland, College Park, Maryland 20742.

[1] Throughout this paper let $\ln = \log_e$ and $\lg = \log_2$.

$\varepsilon 2^{1/\varepsilon}$. This constant factor diverges as ε approaches 0, which is the interesting case in which the reversible time approaches the irreversible time. Using recurrence equations we also prove that Bennett's revised simulation actually runs in time $\Theta(T^{1+\varepsilon}/S^{\varepsilon})$ and space $\Theta(S(1+\ln(T/S)))$.

Bennett achieves his improved simulation by embedding his 1973 technique in a clever general-purpose time-space tradeoff. Although unknown to Bennett, this general-purpose time-space tradeoff was independently discovered by Chandra [5] in 1972.[2]

The rest of this note is divided into two sections. In § 2 we give a simplified analysis of Bennett's time-space tradeoff using recurrence equations. We also plot the tradeoff curve for the reversible time and space for several values of T/S. In § 3 we compute the constant factor in the space bound and correct an error in Bennett's paper.

2. Time and space analysis using recurrence equations. Consider any irreversible program that runs in time T and space S. We assume that $T \geqq 2S$, since otherwise it would be better to use Bennett's 1973 algorithm to produce an equivalent reversible program that runs in time $\Theta(T)$ and space $\Theta(S)$. Bennett's 1988 tradeoff algorithm depends on three integral parameters $m \geqq 1$, $k \geqq 2$, and $n \geqq 0$. The tradeoff algorithm reversibly simulates the irreversible program in segments of m steps using Bennett's 1973 algorithm. For $n > 0$, a total of k^n m-step segments are simulated by performing k forward and $k-1$ backward simulations each consisting of mk^{n-1} steps. For $n = 0$, the 1973 algorithm is used. At the end of each forward simulation only the final configuration is stored. At the end of each backward simulation one previously stored configuration is erased. Using Bennett's notation, let $R(z, x, n, m, d)$ represent the reversible simulation of mk^n steps of the original irreversible program from configuration z to configuration x in segments of size m. The parameter d, which takes on the values 1 and -1, signals whether a forward or backward simulation is taking place.

We now consider the associated recurrence equations for the time and space complexity of the simulation. Let P_n and Q_n be the number of steps in $R(z, x, n, m, d)$ with $d = 1$ and with $d = -1$, respectively. We then have the following coupled recurrence equations

$$P_n = \begin{cases} kP_{n-1} + (k-1)Q_{n-1} & \text{if } n > 0 \\ m & \text{if } n = 0 \end{cases} \tag{1}$$

and

$$Q_n = \begin{cases} kQ_{n-1} + (k-1)P_{n-1} & \text{if } n > 0 \\ m & \text{if } n = 0. \end{cases} \tag{2}$$

Substituting (1) into (2) yields the second-order recurrence

$$P_n = \begin{cases} 2kP_{n-1} - (2k-1)P_{n-2} & \text{if } n > 1 \\ m(2k-1) & \text{if } n = 1 \\ m & \text{if } n = 0, \end{cases} \tag{3}$$

which can be solved exactly by characteristic equations [4] to obtain

$$P_n = m(2k-1)^n. \tag{4}$$

The space bound S_n satisfies the recurrence equation

$$S_n = \begin{cases} (k-1)m + S_{n-1} & \text{if } n > 1 \\ (k-1)m & \text{if } n = 1, \end{cases} \tag{5}$$

[2] For another interesting application of this tradeoff, see the "cycling known-plaintext attack" against group ciphers by Kaliski, Rivest, and Sherman [7].

which describes the number of stored intermediate configurations. By iteration [6] the exact solution is

$$S_n = mn(k-1). \tag{6}$$

Bennett chooses $m = S$ to ensure that each stored configuration takes at most S space. From this assumption (4) and (6) yield, respectively, the following time bound T' and space bound S' for the reversible program

$$T' = S(2k-1)^n \tag{7}$$

and

$$S' = Sn(k-1). \tag{8}$$

Assuming

$$T = Sk^n, \tag{9}$$

that is, assuming the number of simulated steps Sk^n is equal to the running time of the irreversible program, we have $n = (\ln(T/S))/\ln k$. Thus by appropriate choice of k, the user can select any point along the tradeoff curve

$$\frac{T'}{S'} = \frac{(2k-1)^n}{n(k-1)}. \tag{10}$$

Figure 1 shows the logarithm of this tradeoff curve for several values of T/S.

We now separately express T' and S' in terms of T, S, and k. By (7) and (9),

$$\frac{T'}{T} = \left(\frac{2k-1}{k}\right)^n = (2-(1/k))^{(\ln(T/S))/\ln k} = \left(\frac{T}{S}\right)^{\varepsilon(k)} \tag{11}$$

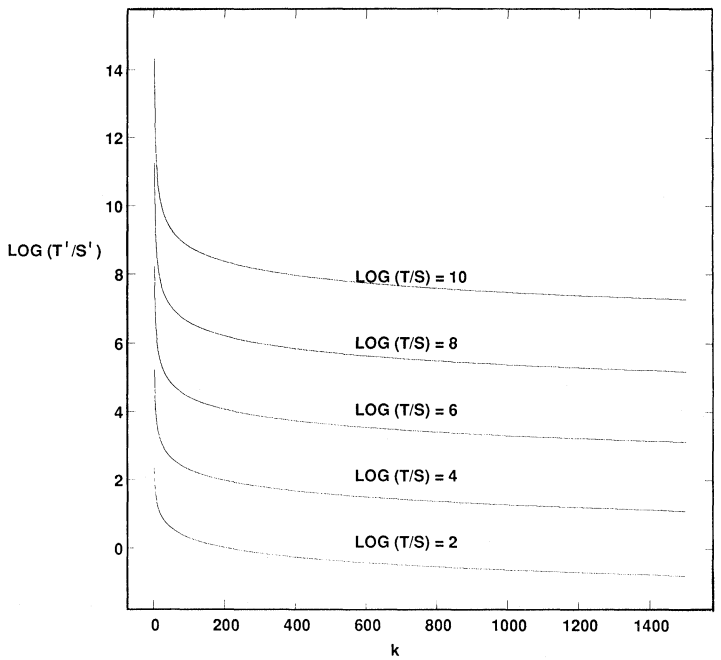

FIG. 1. *Tradeoff curves* $\log_{10}(T'/S')$ *for the reversible time and space as a function of the parameter k and the ratio* T/S *of the irreversible time and space. The curves are drawn for* $T/S = 10^2, 10^4, 10^6, 10^8, 10^{10}$.

and thus

$$T' = T\left(\frac{T}{S}\right)^{\varepsilon(k)} = \frac{T^{1+\varepsilon(k)}}{S^{\varepsilon(k)}} \tag{12}$$

where

$$\varepsilon(k) = \frac{\ln(2-(1/k))}{\ln k}. \tag{13}$$

Similarly by (8),

$$S' = S\frac{\ln(T/S)}{\ln k}(k-1) = c(k)S\ln\frac{T}{S} \tag{14}$$

where

$$c(k) = \frac{k-1}{\ln k}. \tag{15}$$

Therefore, regardless of the relationship between T and S, given any $\varepsilon > 0$ the reversible program can be made to run in time $\Theta(T^{1+\varepsilon}/S^{\varepsilon})$ and space $\Theta(S(1+\ln(T/S)))$. Equations (12) and (14) differ from Bennett's corresponding bounds in two respects. First, our bounds are tight and include the dependence of T' and S' on S.[3] Second, the ε used by Bennett is equal to $1/\lg k$, which is slightly larger and approximately equal to $\varepsilon(k)$ for large k.

3. Calculation of the constant factor in the space bound. The constant factor for the space bound is the term $c(k)$ in (14). To express this factor in terms of $\varepsilon(k)$ we first compute the Taylor series expansion of $\varepsilon(k)$ around $1/k$ to obtain

$$\varepsilon(k) = \frac{1}{\ln k}\left(\ln 2 - \frac{1}{2k} - \frac{1}{8k^2} - \cdots\right). \tag{16}$$

Letting

$$\varepsilon_0(k) = \frac{\ln 2}{\ln k} = \frac{1}{\lg k} \tag{17}$$

be the first term in this expansion (i.e., $\varepsilon_0(k)$ is Bennett's ε), we have $\ln k = (\ln 2)/\varepsilon_0(k)$ and $k = 2^{1/\varepsilon_0(k)}$. Therefore, by (15)

$$c(k) = \frac{\varepsilon_0(k)}{\ln 2}(2^{1/\varepsilon_0(k)} - 1) \sim \frac{\varepsilon_0(k)}{\ln 2}2^{1/\varepsilon_0(k)}. \tag{18}$$

The most interesting case is when $\varepsilon(k)$ tends to 0—that is, when the reversible time approaches the irreversible time—but in this case the term $c(k)$ diverges.

To illustrate how the constant factor $c(k)$ diverges when $\varepsilon(k)$ approaches 0, consider what happens when $aT \leqq T' \leqq bT$ for some constants $1 < a \leqq b$. In this case, the identity $\varepsilon(k) = (\ln(T'/T))/\ln(T/S)$ from (12) implies that

$$\frac{\ln a}{\ln(T/S)} \leqq \varepsilon(k) \leqq \frac{\ln b}{\ln(T/S)}. \tag{19}$$

Note that to be able to find an integer k that satisfies (19), it is necessary for

[3] Although Bennett did not explicitly state the dependence of T' and S' on S, the details of his proof given in the appendix of [1] suggest that he was aware of this dependence.

$a \leqq (T/S)^{\varepsilon_{\max}}$, where $\varepsilon_{\max} = \max\{\varepsilon(k): k \geqq 2\} = \varepsilon(2) \approx 0.58$. Since $k \geqq 2$, (18) implies that

$$\frac{\varepsilon_0(k)}{2 \ln 2} 2^{1/\varepsilon_0(k)} \leqq c(k) < \frac{\varepsilon_0(k)}{\ln 2} 2^{1/\varepsilon_0(k)}. \tag{20}$$

Using the fact that $\varepsilon(k) < \varepsilon_0(k) < 2\varepsilon(k)$, it follows that

$$\frac{\varepsilon(k)}{2 \ln 2} 2^{1/(2\varepsilon(k))} < c(k) < \frac{2\varepsilon(k)}{\ln 2} 2^{1/\varepsilon(k)}. \tag{21}$$

By (19),

$$\frac{(\ln a)/\ln(T/S)}{2 \ln 2} 2^{(\ln(T/S))/(2 \ln b)} < c(k) < \frac{(2 \ln b)/\ln(T/S)}{\ln 2} 2^{(\ln(T/S))/\ln a} \tag{22}$$

and hence

$$\frac{\lg a}{2 \ln(T/S)} \left(\frac{T}{S}\right)^{1/(2 \lg b)} < c(k) < \frac{2 \lg b}{\ln(T/S)} \left(\frac{T}{S}\right)^{1/\lg a}. \tag{23}$$

Equation (14) then yields the space bounds

$$\frac{\lg a}{2} S \left(\frac{T}{S}\right)^{1/(2 \lg b)} < S' < 2(\lg b) S \left(\frac{T}{S}\right)^{1/\lg a}, \tag{24}$$

which grow superlogarithmically in T/S.

We conclude with one more refinement of Bennett's results. Bennett claims that for any $\delta > 0$, the reversible program can be made to run in time $O(T)$ and space $O(ST^{\delta})$. But as Bennett notes, his δ depends on T; hence, Bennett's argument does not hold for arbitrary δ. We correct Bennett's claim as follows. For any $\delta \geqq 1/(\varepsilon_{\max} \lg(T/S))$, we can choose k such that $a = b = 2^{1/\delta}$, in which case (24) yields the tighter space bound

$$\frac{1}{2\delta} S \left(\frac{T}{S}\right)^{\delta/2} < S' < \frac{2}{\delta} S \left(\frac{T}{S}\right)^{\delta}. \tag{25}$$

Equation (25) proves that whenever $T \geqq 2S$ and for any $\delta \geqq 1/(\varepsilon_{\max} \lg(T/S))$, the reversible program can be made to run in time $\Theta(T)$ and space $\Omega(S(T/S)^{\delta/2}) \cap O(S(T/S)^{\delta})$.

Although Bennett's clever time-space tradeoff provides a better way to transform any irreversible program into an equivalent reversible program, its practical utility is diminished by its huge constant factor in the space bound. It remains an open question whether or not Bennett's algorithm yields an optimum tradeoff.

REFERENCES

[1] C. H. BENNETT, *Time/space trade-offs for reversible computation*, SIAM J. Comput., 18 (1989), pp. 766–776.

[2] ———, *Logical reversibility of computation*, IBM J. Res. Devel., 17 (1973), pp. 525–532.

[3] C. H. BENNETT AND R. LANDAUER, *The fundamental physical limits of computation*, Scientific American, 253 (1985), pp. 48–56.

[4] G. BRASSARD AND P. BRATLEY, *Algorithms: Theory & Practice*, Prentice-Hall, Englewood Cliffs, NJ, 1988.

[5] A. K. CHANDRA, *Efficient compilation of linear recursive programs*, Stanford Artificial Intelligence Project Memo AIM-167, STAN-CS-72-282, Computer Science Department, Stanford University, Stanford, CA, April 1972.

[6] T. CORMEN, C. E. LEISERSON, AND R. L. RIVEST, *Introduction to Algorithms*, MIT Press and McGraw-Hill, Cambridge, MA, 1990, to appear.

[7] B. S. KALISKI JR., R. L. RIVEST, AND A. T. SHERMAN, *Is the data encryption standard a group? (Results of cycling experiments on* DES), J. Cryptology, 1 (1988), pp. 3–36.