Computation of pi(x) : improvements to the Meissel, Lehmer, Lagarias, Miller, Odlyzko, Deléglise and Rivat method

Xavier Gourdon

February 15, 2001

Abstract

In 1870, the German astronomer Meissel initiated a method to compute efficiently single values of $\pi(x)$ (the number of primes $\leq x$). His method has been improved several times, first by Lehmer in 1959, then by Lagarias, Miller and Odlyzko in 1985, and finally by Deléglise and Rivat in 1994. We aim at presenting new improvements to this method. As a result, some constant factors are saved in the algorithm, the method is more economical in space, and we present a technique which permits to distribute the computation with minimal memory exchange. Implementation of these improvements permits to reach larger values of x and was used to compute $\pi(10^{21})$ by the author. A distributed project on the web was started and permitted to reach the value of $\pi(2 \times 10^{22})$.

1 Introduction

The Sieve of Eratosthenes was, until 1870, the only known method to compute efficiently $\pi(x)$, when Meissel proposed a method to compute efficiently $\pi(x)$ without computing all the prime numbers less than x. He computed by hand $\pi(10^8)$ and $\pi(10^9)$, reaching values which were far beyond the prime tables at that time. Later in 1959, Lehmer improved the technique and computed $\pi(10^{10})$. In 1985, a breakthrough was obtained when Lagarias, Miller and Odlyzko proved, using refinement of the method, that $\pi(x)$ can be computed in time $O(x^{2/3}/\log(x))$ using $O(x^{1/3}\log^2(x)\log\log(x))$ space. They computed several values of $\pi(x)$ up to 4×10^{16}. Once again, the method was refined in 1994 by Deléglise and Rivat who saved a factor of $\log(x)$ in the time cost, but with a space $O(\log(x))$ bigger. The refinements lead to practical improvement and Deléglise implemented the method so that values up to $x = 10^{20}$ were computed.

We improved again this method, with the following features :

- Constant factors have been saved in the time cost,
- the method is more economical in space,
- the computation can be distributed on several machines with a very small memory exchange, provided a relative cheap precomputation is done on each machine.

As a result, an implementation permitted to reach the value of $\pi(10^{21})$. A distributed version has also been implemented, permitting to obtain from a distributed project on the web the values of $\pi(2 \times 10^{21})$, $\pi(4 \times 10^{21})$, $\pi(10^{22})$ and $\pi(2 \times 10^{22})$. (the distributed project is going on and higher values will be obtained).

Note that in 1987, Lagarias and Odlyzko [3] described an analytic method to compute $\pi(x)$, based on numerical integration involving Riemann ζ-function, using $O(x^{1/2+\epsilon})$ time and $O(x^{1/4+\epsilon})$ space. Despites this asymptotic superiority, the corresponding method has never been implemented and the implied constant are probably large, therefore the method does not seem of practical use for reachable values of x today.

2 Recall of the method

The algorithm is well described in [2] and [1]. We just outline the method. In the following, p and q always denote prime numbers.

Let p_1, p_2, p_3, ... denote the consecutive primes 2, 3, 5, ... We denote by $\phi(x,a)$ the *partial sieve function* which counts numbers $n \leq x$ with all prime factors $> p_a$:

$$\phi(x,a) = \#\{n \leq x; \quad p \mid n \rightarrow p > p_a\}.$$

Let

$$P_k(x,a) = \#\{n \leq x; \quad n = q_1 q_2 \cdots q_k \text{ and } q_1, \ldots, q_k > p_a\},$$

counting numbers $\leq x$ with exactly k prime factors, all larger than p_a. We have the identity

$$\phi(x,a) = P_0(x,a) + P_1(x,a) + \cdots + P_k(x,a) + \cdots$$

where the sum has finitely non zero terms.

We now fix an integer y such that $x^{1/3} \leq y \leq x^{1/2}$, and consider $a = \pi(y)$. We have $P_1(x,a) = \pi(x) - a$ and $P_k(x,a) = 0$ for $k \geq 3$, thus

$$\pi(x) = \phi(x,a) + a - 1 - P_2(x,a).$$

Computation of $P_2(x,a)$

We easily find (see [1] or [2])

$$P_2(x,a) = \sum_{y < p \leq \sqrt{x}} \left(\pi\left(\frac{x}{p}\right) - \pi(p) + 1 \right).$$

Computation of the partial sieve function $\phi(x,a)$

We have the recurrence

$$\phi(x,b) = \phi(x,b-1) - \phi\left(\frac{x}{p_b}, b-1\right). \tag{1}$$

This relation is the basic formula which permits to decrease the complexity of the computation. For example, applying the recurrence (1) at two levels from the value $\phi(x,a)$ leads to

$$\phi(x,a) = \phi(x,a-2) - \phi\left(\frac{x}{p_{a-1}}, a-2\right) - \phi\left(\frac{x}{p_a}, a-2\right) + \phi\left(\frac{x}{p_a p_{a-1}}, a-2\right).$$

This process can be continued until we get terms of the form $\phi(u,0)$, leading to the formula

$$\phi(x,a) = \sum_{1 \leq n \leq x, \gamma(n) \leq y} \mu(n) \left[\frac{x}{n}\right],$$

since $\phi(u,0) = [u]$, where $\mu(n)$ denotes the Möbius function and $\gamma(n)$ denotes the greatest prime factor of n.

Unfortunately, this sums contains too many terms and to make the method effective, the truncation rule is replaced by the following (see [1]) :

Do not split a node $\mu(n)\phi(\frac{x}{n}, b)$ if either of the following holds :

1. *$b = k$ and $n \leq z$*
2. *$n > z$*

where k is a small fixed integer value (for example $k = 1$), and z is a fixed value which satisfy $y \leq z < x^{1/2}$.

This rule gives the equality

$$\phi(x, a) = \phi_0 + \phi_1$$

with

$$\phi_0 = \sum_{n \leq z, \delta(n) > p_k, \gamma(n) \leq y} \mu(n) \phi\left(\frac{x}{n}, k\right) \tag{2}$$

$$\phi_1 = -\sum_{p_k < p < y} \sum_{m: m \leq z < pm, \delta(m) > p, \gamma(m) \leq y} \mu(m) \phi\left(\frac{x}{pm}, \pi(p) - 1\right). \tag{3}$$

The value ϕ_0 is easily computed thanks to the formula

$$\phi(z, k) = [z/P_k]\phi(P_k) + \phi(z \bmod P_k, k), \quad P_k = p_1 \cdots p_k.$$

As for ϕ_1, we let $x^* = \max(x^{1/4}, x/y^2)$ and we have

$$\phi_1 = S_1 + S_2,$$

where

$$S_1 = -\sum_{p_k < p \leq x^*} \sum_{m: m \leq z < pm, \delta(m) > p, \gamma(m) \leq y} \mu(m) \phi\left(\frac{x}{pm}, \pi(p) - 1\right)$$

$$S_2 = \sum_{x^* < p < y} \sum_{q: p < q \leq y} \phi\left(\frac{x}{pq}, \pi(p) - 1\right).$$

This is true since the value of m in (3) for which $p > x^*$ satisfy $p < m < z$ and $\delta(m) > p$, thus $\delta(m)^2 > p^2 > x^{1/2} > z$, thus $m < \delta(m)^2$ thus $m = q$ is prime, and $mp = pq > p^2 > z$.

Computation of S_2

We rewrite S_2 in the form

$$S_2 = T_1 + T_2 + T_3,$$

where

$$T_1 = \sum_{x^* < p \leq (x/y)^{1/2}} \sum_{p < q \leq y} \phi\left(\frac{x}{pq}, \pi(p) - 1\right),$$

$$T_2 = \sum_{(x/y)^{1/2} < p \leq x^{1/3}} \sum_{p < q \leq y} \phi\left(\frac{x}{pq}, \pi(p) - 1\right),$$

$$T_3 = \sum_{x^{1/3} < p < y} \sum_{p < q \leq y} \phi\left(\frac{x}{pq}, \pi(p) - 1\right),$$

In T_3, we always have $x/(pq) < p$ so $\phi(x/(pq), \pi(p) - 1) = 1$ and finally

$$T_3 = \frac{(\pi(y) - \pi(x^{1/3}))(\pi(y) - \pi(x^{1/3}) - 1)}{2}.$$

As for T_2, we distinguish its contribution T_2'' where $x/p^2 < q \leq y$, for which $x/(pq) < p$ thus $\phi(x/(pq), \pi(p) - 1) = 1$. The corresponding value is

$$T_2'' = \sum_{(x/y)^{1/2} < p \leq x^{1/3}} \pi(y) - \pi\left(\frac{x}{p^2}\right).$$

The complementary value $T_2' = T_2 - T_2''$ is

$$T_2' = \sum_{(x/y)^{1/2} < p \le x^{1/3}} \; \sum_{p < q \le x/p^2} \phi\left(\frac{x}{pq}, \pi(p) - 1\right).$$

We now concentrate on the terms T_1 and T_2'. For both, we always have $p \le x/(pq) < p^2$ thus

$$\phi\left(\frac{x}{pq}, \pi(p) - 1\right) = \pi\left(\frac{x}{pq}\right) - \pi(p) + 2.$$

As a consequence, we have

$$T_1 + T_2' = U + V_1 + V_2,$$

where

$$U = \sum_{x^* < p \le x^{1/3}} \; \sum_{p < q \le \min(y, x/p^2)} 2 - \pi(p),$$

and

$$V_1 = \sum_{x^* < p \le (x/y)^{1/2}} \; \sum_{p < q \le y} \pi\left(\frac{x}{pq}\right), \qquad V_2 = \sum_{(x/y)^{1/2} < p \le x^{1/3}} \; \sum_{p < q \le x/p^2} \pi\left(\frac{x}{pq}\right).$$

Decreasing the number of terms

The V_1 and V_2 terms contain a number of terms (p, q) which is asymptotically proportional to $(xy)^{1/2}/\log^2 x$, which is too much. To decrease this number of terms, we split the summations again using the following result. (Note : this results also leads to a simplification of the method presented in [1] to compute W_3.)

Lemma 1

$$\sum_{\alpha < q \le \beta} \pi\left(\frac{z}{q}\right) = \pi\left(\frac{z}{\beta}\right)\pi(\beta) - \pi\left(\frac{z}{\alpha}\right)\pi(\alpha) + \sum_{z/\beta < q \le z/\alpha} \pi\left(\frac{z}{q}\right). \tag{4}$$

Proof : Remember that p_j is the j-th prime number. We have

$$\pi\left(\frac{z}{q}\right) = j \quad \text{iff} \quad \frac{z}{p_{j+1}} < q \le \frac{z}{p_j}.$$

Now we define a and b such that

$$\frac{z}{p_{a+1}} < \alpha \le \frac{z}{p_a} < \frac{z}{p_{a-1}} < \cdots < \frac{z}{p_{b+2}} < \frac{z}{p_{b+1}} < \beta \le \frac{z}{p_b}.$$

Denoting by S the left side of (4) we have

$$S = b\left[\pi(\beta) - \pi\left(\frac{z}{p_{b+1}}\right)\right] + \sum_{j=b+1}^{a-1} j\left[\pi\left(\frac{z}{p_j}\right) - \pi\left(\frac{z}{p_{j+1}}\right)\right] + a\left[\pi\left(\frac{z}{p_a}\right) - \pi(\alpha)\right]$$

thus

$$S = b\pi(\beta) - a\pi(\alpha) + \sum_{j=b+1}^{a} \pi\left(\frac{z}{p_j}\right).$$

The inversion equality (4) follows since $a = \pi(z/\alpha)$ and $b = \pi(z/\beta)$. •

We now use this equality to rewrite V_2 by splitting it for $(x/p)^{1/2} < q$. The equality (4), used with $z = x/p$, $\alpha = (x/p)^{1/2}$ and $\beta = x/p^2$ writes as

$$\sum_{(x/p)^{1/2} < q \le x/p^2} \pi\left(\frac{x}{pq}\right) = \pi(p)\pi\left(\frac{x}{p^2}\right) - \pi\left(\frac{x^{1/2}}{p^{1/2}}\right)^2 + \sum_{p < q \le (x/p)^{1/2}} \pi\left(\frac{x}{pq}\right).$$

This implies

$$V_2 = W_2 + 2 \sum_{(x/y)^{1/2} < p \le x^{1/3}} \sum_{p < q \le (x/p)^{1/2}} \pi\left(\frac{x}{pq}\right), \quad W_2 = \sum_{(x/y)^{1/2} < p \le x^{1/3}} \pi(p)\pi\left(\frac{x}{p^2}\right) - \pi\left(\frac{x^{1/2}}{p^{1/2}}\right)^2 .$$

In the same vein, the use of (4) with $z = x/p$, $\alpha = (x/p)^{1/2}$ and $\beta = y$ leads to

$$V_1 = W_1 + \sum_{x^* < p \le (x/y)^{1/2}} \sum_{p < q \le (x/p)^{1/2}} \chi\left(\frac{x}{pq}\right)\pi\left(\frac{x}{pq}\right), \qquad \chi\left(\frac{x}{pq}\right) = \begin{cases} 2 \text{ if } x/(pq) < y \\ 1 \text{ if } x/(pq) \ge y \end{cases}$$

with

$$W_1 = \sum_{x^* < p \le (x/y)^{1/2}} \pi\left(\frac{x}{py}\right)\pi(y) - \pi\left(\frac{x^{1/2}}{p^{1/2}}\right)^2 .$$

Finally, this simplifies to

$$V_1 + V_2 = X - Y + W_1' + W_2'$$

where

$$X = \sum_{x^* < p \le x^{1/3}} \sum_{p < q \le (x/p)^{1/2}} \chi\left(\frac{x}{pq}\right)\pi\left(\frac{x}{pq}\right), \qquad Y = \sum_{x^* < p \le x^{1/3}} \pi\left(\frac{x^{1/2}}{p^{1/2}}\right)^2 ,$$

and

$$W_1' = \sum_{x^* < p \le (x/y)^{1/2}} \pi\left(\frac{x}{py}\right)\pi(y), \qquad W_2' = \sum_{(x/y)^{1/2} < p \le x^{1/3}} \pi(p)\pi\left(\frac{x}{p^2}\right)$$

Now we go back to U. We distinguish its contribution U_1 for which $x^* < p \le (x/y)^{1/2}$ and its complementary U_2. We have

$$\begin{aligned} U = U_1 + U_2, \qquad U_1 &= \sum_{x^* < p \le (x/y)^{1/2}} (\pi(y) - \pi(p))(2 - \pi(p)) \\ U_2 &= \sum_{(x/y)^{1/2} < p \le x^{1/3}} \left(\pi\left(\frac{x}{p^2}\right) - \pi(p)\right)(2 - \pi(p)). \end{aligned}$$

The prime p in W_2' runs on the same range than U_2 and T_2'', and we have

$$U_2 + W_2' + T_2'' = \sum_{(x/y)^{1/2} < p \le x^{1/3}} \left[\pi\left(\frac{x}{p^2}\right) - 2\pi(p) + \pi(p)^2 + \pi(y)\right].$$

Summing this with $U_1 + W_1'$ gives

$$\begin{aligned} U_1 + W_1' + U_2 + W_2' + T_2'' &= \sum_{x^* < p \le (x/y)^{1/2}} \pi(y)\left[2 - \pi(p) + \pi\left(\frac{x}{py}\right)\right] \\ &+ \sum_{x^* < p \le x^{1/3}} \pi(p)^2 - 2\pi(p) + \sum_{(x/y)^{1/2} < p \le x^{1/3}} \left[\pi\left(\frac{x}{p^2}\right) + \pi(y)\right]. \end{aligned}$$

This is also equal to

$$\begin{aligned}
\sigma &= \pi(y)\left[\pi(x^{1/3}) - \pi((x/y)^{1/2}) - \frac{\pi((x/y)^{1/2})(\pi((x/y)^{1/2})-3)}{2} + \frac{\pi(x^*)(\pi(x^*)-3)}{2}\right] \\
&+ \frac{\pi(x^{1/3})(\pi(x^{1/3})-1)(2\pi(x^{1/3})-1)}{6} - \pi(x^{1/3}) - \frac{\pi(x^*)(\pi(x^*)-1)(2\pi(x^*)-1)}{6} + \pi(x^*) \\
&+ \pi(y) \sum_{x^*<p\le(x/y)^{1/2}} \pi\left(\frac{x}{py}\right) \\
&+ \sum_{(x/y)^{1/2}<p\le x^{1/3}} \pi\left(\frac{x}{p^2}\right).
\end{aligned}$$

Theorem 1 *Let y such that $x^{1/3} < y < x^{1/2}$, z such that $y \le z < x^{1/2}$ and k a fixed small constant. Then we have*

$$\pi(x) = A - B + \omega + \phi_0 + \Sigma,$$

where, using the notation $x^* = \max(x^{1/4}, x/y^2)$,

$$\begin{aligned}
A &= \sum_{x^*<p\le x^{1/3}} \sum_{p<q\le(x/p)^{1/2}} \chi\left(\frac{x}{pq}\right)\pi\left(\frac{x}{pq}\right), \qquad \chi\left(\frac{x}{pq}\right) = \begin{cases} 2 \text{ if } x/(pq) < y \\ 1 \text{ if } x/(pq) \ge y \end{cases} \\
B &= \sum_{y<p\le x^{1/2}} \pi\left(\frac{x}{p}\right) \\
\omega &= -\sum_{p_k<p\le x^*} \sum_{m:m\le z<pm,\delta(m)>p,\gamma(m)\le y} \mu(m)\phi\left(\frac{x}{pm}, \pi(p)-1\right) \\
\phi_0 &= \sum_{n\le y,\delta(n)>p_k} \mu(n)\phi\left(\frac{x}{n}, k\right)
\end{aligned}$$

and $\Sigma = \sum_{i=0}^{6} \Sigma_i$ is the auxilliary term, defined with the notations

$$a = \pi(y), \quad b = \pi(x^{1/3}), \quad c = \pi((x/y)^{1/2}), \quad d = \pi(x^*)$$

by

$$\begin{aligned}
\Sigma_0 &= a - 1 + \frac{\pi(x^{1/2})(\pi(x^{1/2})-1)}{2} - \frac{a(a-1)}{2} \\
\Sigma_1 &= \frac{(a-b)(a-b-1)}{2} \\
\Sigma_2 &= a\left[b - c - \frac{c(c-3)}{2} + \frac{d(d-3)}{2}\right] \\
\Sigma_3 &= \frac{b(b-1)(2b-1)}{6} - b - \frac{d(d-1)(2d-1)}{6} + d \\
\Sigma_4 &= \pi(y) \sum_{x^*<p\le(x/y)^{1/2}} \pi\left(\frac{x}{py}\right) \\
\Sigma_5 &= \sum_{(x/y)^{1/2}<p\le x^{1/3}} \pi\left(\frac{x}{p^2}\right) \\
\Sigma_6 &= -\sum_{x^*<p\le x^{1/3}} \pi\left(\frac{x^{1/2}}{p^{1/2}}\right)^2.
\end{aligned}$$

Algorithm

To accelerate the computation of ω, we use the relation

$$p \le \frac{x}{pm} < p^2 \quad \Longrightarrow \quad \phi\left(\frac{x}{pm}, \pi(p)-1\right) = \pi\left(\frac{x}{pm}\right) - \pi(m) + 2.$$

More precisely, we write $\omega = C + D$ where C contains the terms of omega for which $x/(pm) < p^2$.

Finally, the algorithm computes the value

$$\pi(x) = A - B + C + D + \phi_0 + \Sigma, \tag{5}$$

where

$$\begin{aligned}
A &= \sum_{x^* < p \le x^{1/3}} \sum_{p<q\le (x/p)^{1/2}} \chi\left(\frac{x}{pq}\right)\pi\left(\frac{x}{pq}\right), \qquad \chi\left(\frac{x}{pq}\right) = \begin{cases} 2 \text{ if } x/(pq) < y \\ 1 \text{ if } x/(pq) \ge y \end{cases} \\
B &= \sum_{y<p\le x^{1/2}} \pi\left(\frac{x}{p}\right) \\
C &= -\sum_{p_k<p\le x^*} \sum_{m: m\le z<pm, \delta(m)>p, \gamma(m)\le y, m>x/p^3} \mu(m)\left(\pi\left(\frac{x}{pm}\right) - \pi(p) + 2\right) \\
D &= -\sum_{p_k<p\le x^*} \sum_{m: m\le z<pm, \delta(m)>p, \gamma(m)\le y, m\le x/p^3} \mu(m)\phi\left(\frac{x}{pm}, \pi(p)-1\right) \\
\phi_0 &= \sum_{n\le y, \delta(n)>p_k} \mu(n)\phi\left(\frac{x}{n}, k\right)
\end{aligned}$$

The way these formulaes are used to get the algorithm is not easy and is widely presented in [2] and [1].

As shown in [1], the global resulting cost is optimal when $y \sim cx^{1/3}\log^3 x \log\log x$, with c a positive constant, leading to a global cost of $O(x^{2/3}/\log^2 x)$. The parameter z is an optimization parameter which is choosen to be equal to dy in the practice, with $d > 1$. Only A, B, C and D are non negligeable parts of the computation.

3 Improvements to the method

In this form, the formula is a little different from the one resulting from [1] and a little easier to implement (essentially due to the introduction of the function χ which avoids the computation of W_3 in [1]).

Now, two improvements were made in the implementation :

Decreasing the memory cost

One of the practical problem arising while implementing the method to reach large values of $\pi(x)$ is the memory cost. In [1] and [2], the amount of memory needed is proportional to y, which is a little too big in the practice, especially for distributed computations where the contributors have machine with a small amount of memory. Our approach consisted in using the same technique as presented in [2], first by sieving by blocks of size $O((x/y)^{1/2})$ (instead of blocks of size $O(y)$). The key problem was the ability to access to the values of $\mu(m)$, $\delta(m)$ and $\gamma(m)$ for $m \le z$. In [2] and [1], these values are stored, leading to a memory storage of $O(y)$. To decrease the memory cost, we just stored those values until a value $M = O((x/y)^{1/2})$, and we used different fast processes to compute the other values of $\mu(m)$, $\delta(m)$ and $\gamma(m)$ when needed. A study can be made which shows that these values for $m > M$ are not needed so often, so that the global cost is asymptotically identical.

Distributing the computation

In [2], a method is presented which permits to distribute the computation of the algorithm. Nevertheless, an exchange of memory of size $O(x^{1/3})$ is needed between the machines, making the corresponding implementation nearly impossible for a web distributed project. Following the approach of [2], we found a solution to this problem, provided a precomputation of cost $O(x^{5/9}/\log^7 x)$ is made on each machine. In the case of the computation of the terms A, B and C of (5), the only needed value for a contribution is a starting value $\pi(w)$ for $w = x/n < x/y$. The corresponding precomputation cost is $O(w^{2/3}) = O(x^{4/9})$. The distributed computation of D is a little more difficult. The sum D is divided in contributions corresponding to values (p, m) for which $T < x/(pm) \le U$, where $(T, U]$ is the contribution range $(T < U \le x/z)$. The key is to be able to compute the values $\phi(T, i)$ for $i \le b = \pi(x^*)$ in a cheap precomputation. Applying the recurrence formula (1), we have

$$\begin{array}{rcl} \phi(T,k+1) & = & \phi(T,k) - \phi(T/p_k,k) \\ \phi(T,k+2) & = & \phi(T,k+1) - \phi(T/p_{k+1},k+1) \\ \dots & \dots & \dots \\ \phi(T,b-1) & = & \phi(T,b-2) - \phi(T/p_{b-2},b-2) \\ \phi(T,b) & = & \phi(T,b-1) - \phi(T/p_{b-1},b-1) \end{array}$$

Thus we only need to compute the values $\phi(T/p_k, k)$, $\phi(T/p_{k+1}, k+1)$, ..., $\phi(T/p_{b-1}, b-1)$. Using the Meissel, Lehmer, Lagarias, Miller, Odlyzko, Deléglise and Rivat method again (one recursive level), the cost to compute these values is proportional to

$$\frac{1}{\log^2(T)}\left((T/p_k)^{2/3} + (T/p_{k+1})^{2/3} + \cdots + (T/p_{b-1})^{2/3}\right) = \frac{T^{2/3}}{\log^2 T}\sum_{i=k}^{b-1}\frac{1}{p_i^{2/3}} = O\left(\frac{x^{5/9}}{\log^7 x}\right).$$

References

[1] M. Deléglise and J. Rivat, *Computing $\pi(x)$: the Meissel, Lehmer, Lagarias, Miller, Odlyzko method*, Math. Comp., 65 (1996) 235-245

[2] J. C. Lagarias, V. S. Miller and A. M. Odlyzko, *Computing $\pi(x)$: the Meissel-Lehmer method*, Math. Comp., 44 (1985) 537-560.

[3] J. C. Lagarias and A. M. Odlyzko, *Computing $\pi(x)$: An Analytic Method*, Journal of Algorithms, 8:173-191, 1987.