Информатика

УДК 519.677

ВЫЧИСЛЕНИЕ ГИПЕРГЕОМЕТРИЧЕСКИХ РЯДОВ С КВАЗИЛИНЕЙНОЙ ВРЕМЕННОЙ И ЛИНЕЙНОЙ ЁМКОСТНОЙ СЛОЖНОСТЬЮ

С. В. Яхонтов

Санкт-Петербургский государственный университет, математико-механический факультет, 198504, Санкт-Петербург, Старый Петергоф, Университетский пр-т, 28.

E-mails: sergey_home_mail@inbox.ru

Проводится построение простого для практической реализации алгоритма со сложностью $\mathsf{O}(M(n)\log(n)^2)$ по времени и $\mathsf{O}(n)$ по памяти для вычисления гипергеометрических рядов с рациональными коэффициентами на машине Шёнхаге, где $M(n)$ — сложность умножения целых чисел. Показывается, что данный алгоритм пригоден в практической информатике для построения конструктивных аналогов часто используемых констант математического анализа.

Ключевые слова: *конструктивные вещественные числа, гипергеометрические ряды, квазилинейная временная сложность, линейная ёмкостная сложность.*

Введение. Предлагается простой для практической реализации алгоритм, квазилинейный по времени и линейный по памяти, на машине Шёнхаге [1], для вычисления приближённых значений гипергеометрических рядов с рациональными коэффициентами. Данные ряды используются для расчёта некоторых констант математического анализа и элементарных функций в рациональных точках.

С помощью $Sch(FQLINTIME//LINSPACE)$ будем обозначать класс алгоритмов, вычислимых на машине Шёнхаге, квазилинейных по времени и линейных по памяти. Основной особенностью машины Шёнхаге является возможность рекурсивных вызовов процедур. Под квазилинейностью понимается ограниченность сверху функцией вида $\mathsf{O}(n\log(n)^k)$ при некотором k.

Отличительной особенностью алгоритмов, основанных на разложениях в ряды, является относительная простота как самих алгоритмов, так и анализа их вычислительной сложности, так как при этом обычно производятся только арифметические операции над рациональными числами без обращения к дополнительным алгоритмам. Для вычислений с небольшим количеством знаков после двоичной точки ряды эффективнее других способов в связи с небольшими константами в оценках вычислительной сложности. Поэтому такие алгоритмы важны в информатике для практических приложений.

Известно [2], что линейно сходящиеся гипергеометрические ряды с рациональными коэффициентами можно вычислять с помощью метода двоичного деления с временной сложностью $\mathsf{O}(M(n)(\log(n))^2)$ и ёмкостной слож-

Яхонтов Сергей Викторович (к.ф.-м.н.), доцент, каф. информатики.

ностью $\mathsf{O}(n\log(n))$ ($M(n)$ — сложность умножения n-битовых целых чисел). В последнее время появились работы, например [3], в которых описываются алгоритмы (на основе модифицированного метода двоичного деления) вычисления значений линейно сходящихся гипергеометрических рядов с временной сложностью $\mathsf{O}(M(n)(\log(n))^2)$ и ёмкостной сложностью $\mathsf{O}(n)$. В [4] предложены простые для практической реализации алгоритмы для вычисления степенных рядов с полиномиальной временной и линейной ёмкостной сложностью.

Недостатком метода из [3] является относительная сложность практической реализации. В настоящей работе предлагается простой для практической реализации алгоритм, который, так же как и алгоритм из [3], является квазилинейным по времени и линейным по памяти. Идея работать с некоторой заданной точностью для того, чтобы вычислить значение гипергеометрического ряда с линейной ёмкостной сложностью, предложена в [5].

Вычислительная сложность конструктивных вещественных чисел и функций подробно освещается в монографии [6]. Множество конструктивных вещественных чисел с квазилинейной по времени и линейной по памяти сложностью вычисления двоично-рациональных приближений будем обозначать $Sch(FQLINTIME//LINSPACE)_{CF}$ (данное множество можно определить по аналогии с множеством $FLINSPACE$ вычислимых действительных чисел [4] с учётом другой вычислительной модели).

Везде далее через n будет обозначаться длина записи точности вычисления 2^{-n} двоично-рациональных приближений. Под функцией $\log(k)$ будем понимать логарифм по основанию 2.

1. Метод двоичного деления. Данный метод предназначен для вычисления значений линейно сходящихся рядов с рациональными коэффициентами, в частности, для вычисления гипергеометрических рядов вида

$$S=\sum_{i=0}^{\infty}\frac{a(i)}{b(i)}\prod_{j=0}^{i}\frac{p(j)}{q(j)}, \tag{1}$$

где a, b, p, q — полиномы с целыми коэффициентами; данный ряд линейно сходится, если его частичная сумма

$$S(\mu(k))=\sum_{i=0}^{\mu(k)}\frac{a(i)}{b(i)}\prod_{j=0}^{i}\frac{p(j)}{q(j)}, \tag{2}$$

где $\mu(k)$ — линейная функция от k, отличается от точного значения не более чем на 2^{-k}: $|S-S(\mu(k))|\leqslant 2^{-k}$.

В классическом варианте метод двоичного деления состоит в следующем. Обозначим $k_1=\mu(k)$. Рассмотрим частичную сумму (2) для некоторых чисел i_1 и i_2, $0\leqslant i_1\leqslant k_1$, $0\leqslant i_2\leqslant k_1$, $i_1\leqslant i_2$:

$$S(i_1,i_2)=\sum_{i=i_1}^{i_2}\frac{a(i)p(i_1)\dots p(i)}{b(i)q(i_1)\dots q(i)}. \tag{3}$$

Будем вычислять величины $P(i_1,i_2)=p(i_1)\dots p(i_2)$, $Q(i_1,i_2)=q(i_1)\dots q(i_2)$, $B(i_1,i_2)=b(i_1)\dots b(i_2)$ и $T(i_1,i_2)=B(i_1,i_2)Q(i_1,i_2)S(i_1,i_2)$. Если $i_1=i_2$, то величины вычисляются напрямую. Иначе ряд делится на две части, левую и правую, и величины $P(i_1,i_2)$, $Q(i_1,i_2)$, $B(i_1,i_2)$ вычисляются для каждой из частей рекурсивно. Затем полученные величины комбинируются:

$$\begin{aligned} &P(i_1,i_2)=P_lP_r, \quad Q(i_1,i_2)=Q_lQ_r, \quad B(i_1,i_2)=B_lB_r,\\ &T(i_1,i_2)=B_rQ_rT_l+B_lP_lT_r. \end{aligned} \tag{4}$$

Алгоритм начинает свою работу с $i_1=0$, $i_2=k_1$. После вычисления величин $T(0,k_1)$, $B(0,k_1)$, $Q(0,k_1)$ осуществляется деление $T(0,k_1)$ на $B(0,k_1)Q(0,k_1)$, чтобы получить результат с заданной точностью. Длины чисел $T(0,k_1)$ и $B(0,k_1)Q(0,k_1)$ пропорциональны $k\log(k)$, то есть алгоритм двоичного деления является квазилинейным по памяти; временная сложность данного алгоритма — $\mathsf{O}(M(k)\log(k)^2)$ [2].

2. Алгоритм вычисления гипергеометрических рядов из [3]. В алгоритме из [3] вычисляются редуцированные числитель и знаменатель дроби $\widehat{T}/\widehat{Q}$, где $\widehat{T}=TBQ$, $\widehat{Q}=BQ$, имеющие длину $\mathsf{O}(n)$, тогда как в классическом варианте вычисляются числитель и знаменатель длины $\mathsf{O}(n\log(n))$. Затем производится деление редуцированного числителя на редуцированный знаменатель с тем, чтобы получить значение суммы ряда с заданной точностью. Параметром усовершенствованного алгоритма из [3] является модуль m — верхняя граница длины редуцированных $\widehat{T}$ и $\widehat{Q}$; модуль m вычисляется как произведение достаточного числа простых чисел. Ряд разбивается на N/G групп, где G — величина, зависящая от m, и для каждой группы применяется классический алгоритм двоичного деления. Далее вычисленные значения комбинируются по модулю m с использованием реконструкции рациональных чисел с тем, чтобы получить $\widehat{T}$ и $\widehat{Q}$.

Недостатком данного алгоритма является необходимость оценивать величину m для каждого вида ряда отдельно; в [3] для этого предлагается достаточно трудоемкий метод, на примере которого оценивается m для ряда $\zeta(3)$. Недостатком алгоритма из [3] является также необходимость реализовывать достаточно сложный алгоритм реконструкции рациональных чисел (rational number reconstruction), имеющий временную сложность $\mathsf{O}(M(n)\log(n))$ и ёмкостную сложность $\mathsf{O}(n)$.

3. Основной алгоритм класса $\boldsymbol{Sch(FQLINTIME//LINSPACE)}$. Модифицируем метод двоичного деления для гипергеометрических рядов так, чтобы алгоритм получился простой и при этом вычисления находились в пределах класса $Sch(FQLINTIME//LINSPACE)$.

Пусть требуется вычислить значения гипергеометрического ряда (1) с точностью 2^{-n}. Для этого достаточно вычислить частичную сумму (2) с точностью $2^{-(n+1)}$, так как

$$\begin{aligned} |S-S(\mu(n+1))^*| \leqslant |S-S(\mu(n+1))| + |S(\mu(n+1)) - S(\mu(n+1))^*| \leqslant \\ \leqslant 2^{-(n+1)} + 2^{-(n+1)} = 2^{-n}. \end{aligned}$$

Здесь $S(\mu(n+1))^*$ — обозначение для приближенного значения $S(\mu(n+1))$. Обозначим $r=\mu(n+1)$. Возьмем минимальную величину k_1 такую, что $2^{k_1}\geqslant r$; пусть $r_1=\lceil r/k_1\rceil$. Запишем частичную сумму (2) в виде

$$P(r)=\sigma_1+\tau_2\left[\sigma_2+\tau_3\big[\sigma_3+\ldots+\tau_{k_1-1}[\sigma_{k_1-1}+\tau_{k_1}\sigma_{k_1}]\big]\right], \tag{5}$$

где $\sigma_1 = \sum_{i=0}^{r_1-1} \frac{a(i)}{b(i)} \prod_{j=0}^{i} \frac{p(j)}{q(j)} = \sigma_1'$,

$$\tau_2 = \prod_{j=0}^{r_1} \frac{p(j)}{q(j)}, \quad \sigma_2 = \frac{a(r_1)}{b(r_1)} + \sum_{i=r_1+1}^{2r_1-1} \frac{a(i)}{b(i)} \prod_{j=r_1+1}^{i} \frac{p(j)}{q(j)} = \xi_2 + \sigma_2',$$

$$\tau_3 = \prod_{j=r_1+1}^{2r_1} \frac{p(j)}{q(j)}, \quad \sigma_3 = \frac{a(2r_1)}{b(2r_1)} + \sum_{i=2r_1+1}^{3r_1-1} \frac{a(i)}{b(i)} \prod_{j=2r_1+1}^{i} \frac{p(j)}{q(j)} = \xi_3 + \sigma_3', \quad \cdots$$

Величины σ_t' будем вычислять классическим методом двоичного деления для суммы (3), где $i_1 = (t-1)r_1+1$, $i_2 = t \cdot r_1 - 1$; для σ_1' возьмем $i_1 = 0$, $i_2 = r_1 - 1$.

Введём следующие обозначения: $\omega = l(W) + 1$, $W = \max(A, B)$, где $A = \max\limits_{i=0..r}(|a(i)|, |b(i)|)$, $B = \max\limits_{j=0..r}(|p(j)|, |q(j)|)$, $l(u)$ — длина битового представления u. Оформим оценку вычислительной сложности расчёта σ_t и τ_t в виде двух лемм.

Лемма 1. *Временная сложность алгоритма двоичного деления для вычисления σ_t ограничена сверху $\mathsf{O}(M(r)\log(r))$; ёмкостная сложность ограничена сверху $\mathsf{O}(r)$.*

Доказательство. Рассмотрим произвольную максимальную цепочку рекурсивных вызовов, полученных в результате вычислений в соответствии с формулами (4). Условимся, что нумерация в цепочке начинается с самого глубокого элемента цепочки: $i = 1, \ldots, \varsigma$, где ς — длина цепочки, $\varsigma \leqslant \lceil \log(2r_1) \rceil$.

Пользуясь методом математической индукции по i, покажем, что длина представления числа T на глубине i при вычислении σ_t' удовлетворяет соотношению $l(T_i) < 2^{i+1}\omega + 2^i$. При этом заметим, что $l(P_i) \leqslant 2^i\omega$, $l(Q_i) \leqslant 2^i\omega$, $l(B_i) \leqslant 2^i\omega$, так как при увеличении i происходит удвоение длины числа. База индукции: $i = 1$, $l(T_1) \leqslant 2\omega < 2^2\omega + 2^1$; индукционный переход:

$$l(T_{i+1}) < 2^i\omega + 2^i\omega + 2^{i+1}\omega + 2^i + 1 < 2^{(i+1)+1}\omega + 2^{i+1}.$$

Отметим, что исходя из этой оценки T_ς имеет длину $O(r)$, так как

$$l(T_\varsigma) < C_1 2^\varsigma \omega \leqslant C_2 \frac{r}{\log(r)} \log(r) = C_2 r;$$

здесь учитывается тот факт, что все коэффициенты $a(i)$, $b(i)$, $p(j)$, $q(j)$ являются полиномами.

Оценим временную сложность вычисления σ_t'. При этом будем учитывать свойство функции сложности умножения: $2M(2^{-1}m) \leqslant M(m)$ (квазилинейные и полиномиальные функции удовлетворяют этому свойству). Так как в узле дерева вызовов на уровне i производится C_3 умножений $2^{\varsigma-i}$ чисел, длины которых не превосходят $2^{i+1}\omega + 2^i$, получаем следующую оценку количества операций, требуемых для вычисления σ_t':

$$\mathrm{Time}(\sigma_t') \leqslant C_3 \sum_{i=1}^{\varsigma} 2^{\varsigma-i} M(2^{i+1}\omega + 2^i) < C_4 \sum_{i=1}^{\varsigma} 2^{\varsigma-i} M(2^{i+2}\omega) =$$

$$= C_4 \sum_{i=1}^{\varsigma} 2^{\varsigma-i} M(2^{\varsigma-(\varsigma-(i+2))}\omega) \leqslant C_5 \sum_{i=1}^{\varsigma} 2^{\varsigma-i} 2^{-\varsigma+(i+2)} M(2^{\varsigma}\omega) \leqslant$$
$$\leqslant C_6 \varsigma M(r) \leqslant C_7 \log(r) M(r)$$

(здесь используется неравенство для $2^{\varsigma}\omega$ из оценки для $l(T_{\varsigma})$). Заключительное деление дает $\mathsf{O}(M(r))$ операций.

Теперь оценим ёмкостную сложность вычисления σ'_t. Учитывая, что в узле дерева вызовов вложенности i количество памяти, расходуемой на временные переменные, — это $C_8(2^{i+1}\omega + 2^i)$, получаем, что количество памяти во всех одновременно существующих рекурсивных вызовах оценивается следующим образом:

$$\mathrm{Space}(\sigma'_t) \leqslant \sum_{i=1}^{\varsigma} C_8(2^{i+1}\omega + 2^i) \leqslant C_9(2^{\varsigma}\omega + 2^{\varsigma}) \leqslant C_{10} r = \mathsf{O}(r). \quad \square$$

Лемма 2. *Временная сложность алгоритма двоичного деления для вычисления числителя и знаменателя* τ_t *ограничена сверху* $\mathsf{O}(M(r)\log(r))$; *ёмкостная сложность ограничена сверху* $\mathsf{O}(r)$.

Доказательство. Оценки вычислительной сложности τ_t совпадают с таковыми для σ_t в силу того, что неравенства из доказательства леммы 1 также актуальны и для τ_t, если вычислять числитель и знаменатель σ_t с помощью алгоритма двоичного деления для произведений. □

Рассчитывать приближённые значения $P(r)^*$ с точностью $2^{-(n+1)}$ по формуле (5) будем в соответствии со следующим итеративным процессом:

$$\begin{aligned} &h_1(m) = \sigma^*_{k_1}, \\ &\widehat{h}_i(m) = \sigma^*_{k_1-i+1} + \tau^*_{k_1-i+2} h_{i-1}, \quad i = 1, \ldots, k_1, \\ &h_i(m) = \widehat{h}_i(m) + \varepsilon_i; \end{aligned} \tag{6}$$

при $i = k_1$ полагаем $P(r)^* = h_{k_1}(m)$. Здесь $m \geqslant r$ (величину m выберем позже); σ^*_i, τ^*_i — приближения σ_i, τ_i с точностью 2^{-m}. Величины $h_i(m)$ получаются отбрасыванием битов $q_{m+1}q_{m+2}\ldots q_{m+j}$ чисел $\widehat{h}_i(m)$ после двоичной точки, начиная с $(m+1)$-го, т. е.

$$|\varepsilon_i| = |h_i(m) - \widehat{h}_i(m)| = 0.0\ldots 0 q_{m+1} q_{m+2} \ldots q_{m+j}, \tag{7}$$

а знак ε_i совпадает со знаком $\widehat{h}_i(m)$ (ясно, что $|\varepsilon_i| < 2^{-m}$).

Предположим, что выполняются следующие условия:

$$|b(i)| \geqslant 2 \text{ для всех } i, \quad |p(j)|/|q(j)| \leqslant 1 \text{ для всех } j. \tag{8}$$

Покажем, что тогда верны следующие две леммы.

Лемма 3. *Для любого* $i = 1, 2, \ldots, k_1$ *справедлива оценка*

$$|h_i(m)| < (i+1) r_1 W. \tag{9}$$

Доказательство. Применим математическую индукцию по j для $h_j(m)$. База индукции при j, равном 1: $|h_1(m)| \leqslant r_1W + 2^{-m} < 2r_1W$. Индукционный переход для $(j+1) \geqslant 2$:

$$|h_{j+1}(m)| = |\sigma^*_{k_1-(j+1)+1} + \tau^*_{k_1-(j+1)+2}h_j + \varepsilon_{j+1}| \leqslant \\ \leqslant \frac{1}{2}r_1W + (j+1)r_1W + 2^{-m} < ((j+1)+1)r_1W. \quad \square$$

Лемма 4. *Погрешность вычисления* $h_{k_1}(m)$ *по схеме* (6) *оценивается как*

$$\Delta(k_1, m) < 2^{-m}mk_1^2W.$$

Доказательство. Обозначим $H_1 = \sigma_{k_1}$, $H_i = \sigma_{k_1-i+1} + \tau_{k_1-i+2}H_{i-1}$, $\eta(i,m) = |h_i(m) - H_i|$. Воспользуемся методом математической индукции для $\eta(j,m)$ по j. База индукции при j, равном 1:

$$\eta(1,m) = |h_1(m) - H_1| = |\sigma^*_{k_1} - \sigma_{k_1}| < 2^{-m}.$$

Индукционный переход для $(j+1) \geqslant 2$:

$$\eta(j+1,m) = |\sigma^*_{k_1-(j+)+1} + \tau^*_{k_1-(j+1)+2}h_j(m) + \varepsilon_{j+1} - \\ - \sigma_{k_1-(j+1)+1} - \tau_{k_1-(j+1)+2}H_j| < \\ < |\tau^*_v h_j(m) - \tau_v h_j(m) + \tau_v h_j(m) - \tau_v H_j| + 2\cdot 2^{-m} \leqslant \\ \leqslant 2^{-m}h_j(m) + \eta(j,m) + 2\cdot 2^{-m}.$$

Так как из (9) $|h_j(m)| < (j+1)r_1W$ и, по индукционному предположению, $\eta(j,m) < 2^{-m}mj^2W$, можно записать

$$\eta(j+1,m) < 2^{-m}(j+1)r_1W + 2^{-m}mj^2W + 2\cdot 2^{-m} < \\ < 2^{-m}(j+1)mW + 2^{-m}m(j+1)^2W = 2^{-m}m(j+2)^2W.$$

Из $\Delta(k_1,m) = \eta(k_1,m)$ получаем искомое неравенство. $\square$

Из леммы 4 следует, что достаточно взять m такое, чтобы выполнялось

$$m \geqslant (n+1) + \lceil 2\log(n+1) + 2\log(r) + \log(W)\rceil, \tag{10}$$

для вычисления $P(r)^*$ с точностью $2^{-(n+1)}$.

Обозначим алгоритм расчёта гипергеометрического ряда, использующий схему (6), через *LinSpaceBinSplit* (linear space binary splitting).

Алгоритм LinSpaceBinSplit. Приближённое значение ряда (1).

`Вход:` *Запись точности вычисления* 2^{-n}.

`Выход:` *Приближённое значение ряда* (1) *с точностью* 2^{-n}.

`Описание:`

1) *вычисляем* $r := \mu(n+1)$; *подбираем* k_1 *так, чтобы* $2^{k_1} \geqslant r$; *вычисляем* $r_1 := \lceil r/k_1\rceil$; *рассчитываем* W;
2) *рассчитываем* m *по формуле* (10);

3) $h := \sigma^*_{k_1}$ *(с помощью обычного алгоритма двоичного деления с точностью* 2^{-m}*);*
4) *выполняем цикл по* i *от* 2 *до* k_1:
   а) *рассчитываем* $v_1 := \sigma^*_{k_1-i+1}$ *с точностью* 2^{-m} *с помощью обычного алгоритма двоичного деления и* $v_2 := \tau^*_{k_1-i+2}$ *с точностью* 2^{-m},
   б) *вычисляем выражение* $\widehat{h} := v_1 + v_2 h$,
   в) h *присваиваем величину* $\widehat{h}$*, округлённую в соответствии с* (7);
5) *на выход записываем* h.

Оценим временную вычислительную сложность данного алгоритма, учитывая, что r и m линейно зависят от n:
– $\mathsf{O}(\log(n))$ вычислений σ_t дают $\mathsf{O}(M(n)\log(n)^2)$;
– $\mathsf{O}(\log(n))$ вычислений τ_t дают $\mathsf{O}(M(n)\log(n)^2)$;
– $\mathsf{O}(\log(n))$ умножений чисел длины $\mathsf{O}(n)$ дают $\mathsf{O}(M(n)\log(n))$;
итого получаем $\mathsf{O}(M(n)\log(n)^2)$ битовых операций. Ёмкостная сложность алгоритма *LinSpaceBinSplit* — $\mathsf{O}(n)$, так как во всех вычислениях в данном алгоритме фигурируют числа длины $\mathsf{O}(n)$.

Теорема. *Модифицированный алгоритм двоичного деления для расчёта гипергеометрических рядов LinSpaceBinSplit принадлежит классу сложности* $Sch(FQLINTIME//LINSPACE)$.

Заключение. В табл. 1, 2 приведены формулы и ряды для расчёта некоторых часто используемых на практике констант математического анализа. Данные ряды линейно сходятся (см. [2–4]), и они удовлетворяют условиям (8); следовательно, для расчёта их приближённых значений можно использовать алгоритм *LinSpaceBinSplit*, то есть перечисленные константы принадлежат множеству конструктивных чисел $Sch(FQLINTIME//LINSPACE)_{CF}$. Алгоритм *LinSpaceBinSplit* можно также использовать для вычисления приближений многих других констант и приближений элементарных функций в рациональных точках.

Если для умножения использовать алгоритм Шёнхаге—Штрассена с временной сложностью $\mathsf{O}(n\log(n)\log\log(n))$, то временная сложность алгоритма *LinSpaceBinSplit* будет $\mathsf{O}(n\log(n)^3\log\log(n))$; при использовании для умножения простого рекурсивного метода с временной сложностью $\mathsf{O}(n^{\log(3)})$ временная сложность алгоритма *LinSpaceBinSplit* будет $\mathsf{O}(n^{\log(3)}\log(n)^2)$.

Отметим, что ряд константы e сходится со скоростью $2^{-\mathsf{O}(n\log(n))}$, поэтому временная сложность расчёта e в соответствии с алгоритмом *LinSpaceBinSplit* будет $\mathsf{O}(M(n)\log(n))$, а ёмкостная — $\mathsf{O}\left(n/\log(n)\right)$.

Таблица 1

Формулы для расчёта констант

Константы	Формулы	Ряды
e	$e = \exp(1)$	$\exp(1) = 2\sum_{i=0}^{\infty}\frac{1}{2i!}$
π	$\pi = 16\alpha - 4\beta$	$\alpha = \operatorname{arctg}\left(\frac{1}{5}\right) = 2\frac{1}{5}\sum_{i=0}^{\infty}(-1)^i\frac{1}{2(2i+1)5^{2i}}$, $\beta = \operatorname{arctg}\left(\frac{1}{239}\right) = 2\frac{1}{239}\sum_{i=0}^{\infty}(-1)^i\frac{1}{2(2i+1)239^{2i}}$
$\zeta(3)$		$\sum_{i=0}^{\infty}\frac{(-1)^i(205i^2+250i+77)((i+1)!)^5(i!)^5}{2((2i+2)!)^5}$

Таблица 2

Ряды для расчёта констант

Константы	$a(i)$	$b(i)$	$p(j)$	$q(j)$
e	1	2	1	j
π	1	$2(2i+1)$	-1	5^2
	1	$2(2i+1)$	-1	239^2
$\zeta(3)$	$205i^2+250i+77$	2	$p(0)=1$, $p(j)=-j^5$	$32(2j+1)^5$

Из дальнейших исследований можно отметить построение алгоритмов расчёта гипергеометрических рядов с рациональными коэффициентами с временной сложностью $\mathsf{O}(n\log(n)^k)$, $k\leqslant 3$, и при этом имеющих линейную ёмкостную вычислительную сложность.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Schönhage A., Grotefeld A. F. W, Vetter E.* Fast algorithms. A multitape Turing machine implementation. Mannheim, Germany: BI-Wissenschaftsverlag, 1994. 298 pp.
2. *Haible B., Papanikolaou T.* Fast multiprecision evaluation of series of rational numbers / In: *Algorithmic number theory*: Proceedings of 3rd international symposium (Portland, OR, 1998) / Lect. Notes Comput. Sci., 1423. Berlin: Springer, 1998. Pp. 338–350.
3. *Cheng H., Gergel B., Kim E., Zima E.* Space-efficient evaluation of hypergeometric series // *SIGSAM Bull.*, 2005. Vol. 39, no. 2. Pp. 41–52.
4. *Yakhontov S. V.* FLINSPACE constructive real numbers and functions (in Russian). Saarbrücken, Germany: Lambert Academic Publishing, 2010. 167 pp.
5. *Gourdon X., Sebah P.* Binary splitting method: `http://numbers.computation.free.fr/Constants/Algorithms/splitting.ps`.
6. *Ko K.-I* Complexity theory of real functions. Progress in Theoretical Computer Science. Boston, MA: Birkhäuser Boston, Inc., 1991. 310 pp.

Поступила в редакцию 01/II/2011;
в окончательном варианте — 24/VIII/2011.

MSC: 03F60; 68Q17

CALCULATION OF HYPERGEOMETRIC SERIES WITH QUASI-LINEAR TIME AND LINEAR SPACE COMPLEXITY

S. V. Yakhontov

St. Petersburg State University, Mathematics and Mechanics Faculty,
28, Universitetskiy prosp., Stariy Petergoff, St. Petersburg, 198504.

E-mails: `sergey_home_mail@inbox.ru`

A simple for practical implementation algorithm with the time complexity $\mathsf{O}(M(n)\log(n)^2)$ *and space complexity* $\mathsf{O}(n)$ *for the evaluation of hypergeometric series with rational coefficients on the Schönhage machine is constructed (here* $M(n)$ *is the complexity of integer multiplication). It is shown that this algorithm is suitable in practical informatics for constructive analogues of often used constants of analysis.*

Key words: *constructive real numbers, hypergeometric series, quasi-linear time, linear space complexity.*

Original article submitted 01/II/2011;
revision submitted 24/VIII/2011.

Sergey V. Yakhontov (Ph. D. (Phys. & Math.)), Associate Professor, Dept. of Informatics.