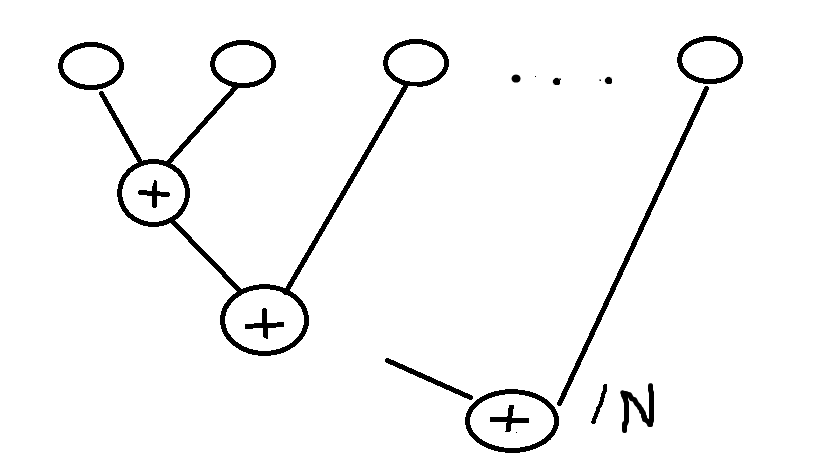
**Лабораторная работа №4**

**Алгоритм:**  Вычисление среднего арифметического N чисел.

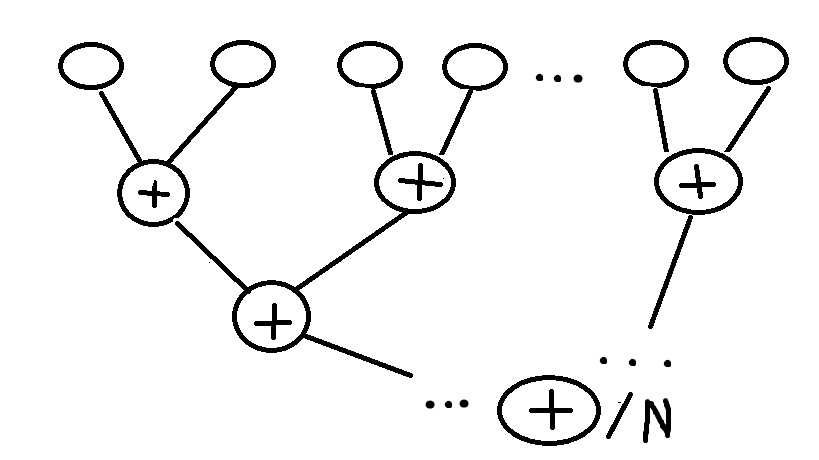
1. Все числа складываются. (n-1) раз
2. Полученная сумма делится на количество чисел.



**Сложность** **последовательной** реализации алгоритма О(n).

**Для распараллеливания** воспользуемся каскадной схемой:

1. На первой итерации все исходные данные разбиваются на пары, для каждой пары вычисляется сумма их значений .
2. Далее все полученные суммы также разбиваются на пары, и снова выполняется суммирование.
3. Конечная сумма делится на количество всех чисел.



**Определим ускорение и эффективность:**

***Ускорение****,*  получаемое при использовании параллельного алгоритма для p процессоров, по сравнению с последовательным вариантом выполнения вычислений определяется величиной:

S_p\left(n\right)=T_1\left(n\right)/T_p\left(n\right)

Число операций суммирования в последовательной схеме: Кпослед = (n-1)

Число параллельных операций суммирования (общее число такое же как и в параллельной): Ккаскад=log2n

Считаем, что время выполнения операций в обоих схемах одинакого, тогда:

Sp(n) = (n-1)/log2n

Количестве процессоров , необходимое для выполнения каскадной схемы p ≥ (n/2)

***Эффективность***использования параллельным алгоритмом процессоров при решении задачи определяется соотношением:

E_p\left(n\right)=T_1\left(n\right)/\left(pT_p\left(n\right)\right)=S_p\left(n\right)/p

В данном случае получим:

Ep(n) = (n-1)/(p\*log2n)

**Предельные значения ускорения и эффективности алгоритма:**

Хотя ускорение увеличивается при увеличении размерности задачи, эффективность стремится к 0.

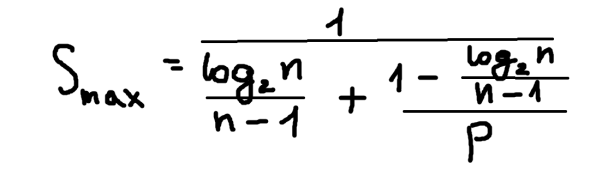
**Для алгоритма каскадного суммирования найдем максимальное ускорение по закону Амдала.**

Время выполнения последовательной части алгоритма = log2n;

Время выполнения всего алгоритма на одном процессоре = n-1;

F - часть времени, не поддающуюся распараллеливанию, в общем времени счёта исходной последовательной программы.

f = log2n / (n-1) =>



**Теорема**. Пусть для некоторой вершины вывода в вычислительной схеме алгоритма существует путь из каждой вершины ввода. Кроме того, пусть входная степень вершин схемы (количество входящих дуг) не превышает 2. Тогда минимально возможное время выполнения параллельного алгоритма ограничено снизу значением

T_\infty\left(G\right)=\log_2n

**Теорема**. Времени выполнения алгоритма, которое сопоставимо с минимально возможным временем T_\infty , можно достичь при количестве процессоров порядка p\sim \frac{T_1}{T_\infty}.

В нашем случае: p>= (n-1)/log2n

p\geq T_1/ T_\infty\Longrightarrow T_p\leq 2T_\infty

Т.е. Tp <= 2 log2n;

Найдем ускорение: Sp =

Найдем эффективность: Ep =

