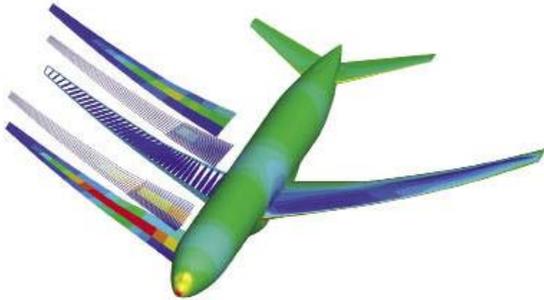


# Программная инфраструктура для решения задач глобальной оптимизации на суперкомпьютерах

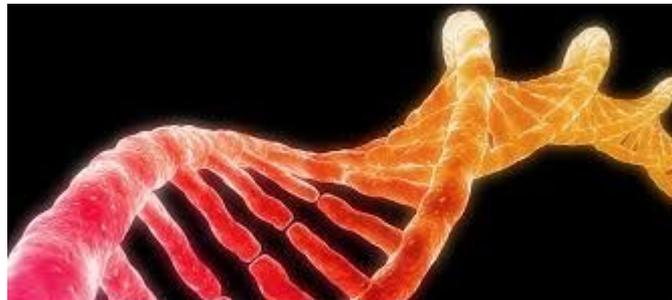
Евтушенко Ю.Г., Голубева Я.В.,  
Посыпкин М.А., Орлов Ю.В.

Вычислительный центр им. А.А.  
Дородницына ФИЦ ИУ РАН

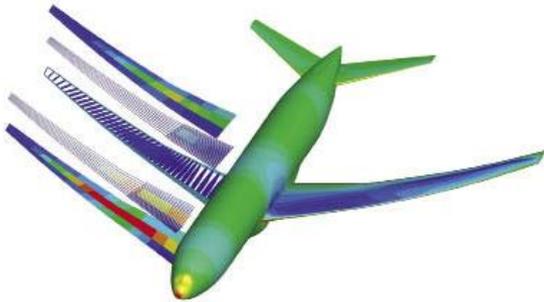
# Задачи глобальной оптимизации



- Инженерная оптимизация
- Логистика
- Нанотехнологии
- Био-информатика
- Математическая экономика



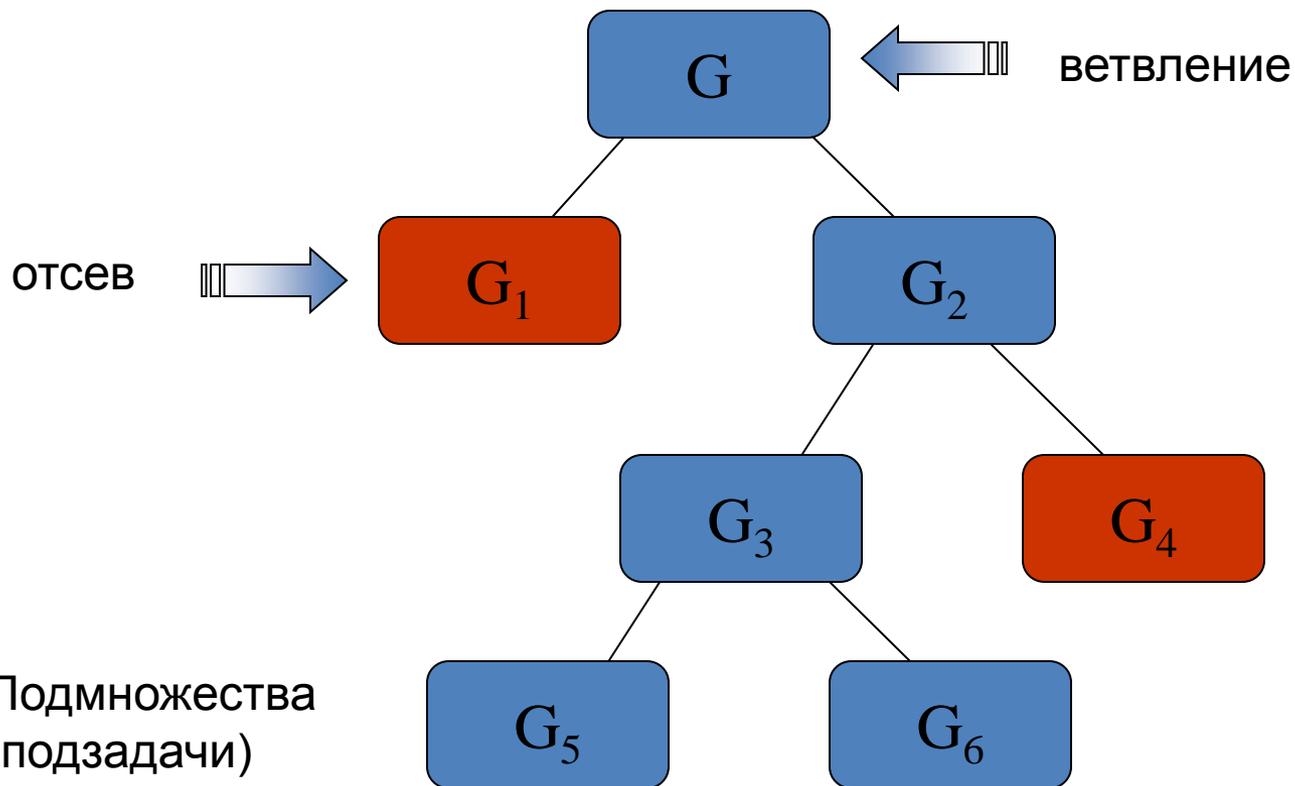
# Методы глобальной оптимизации



- Локальные методы
- Глобальные методы
  - Недетерминированные методы (оценка не дается, либо носит вероятностный характер)
  - **Детерминированные методы** (дается оценка отклонения найденного значения от оптимума)



# Схема ветвей и границ

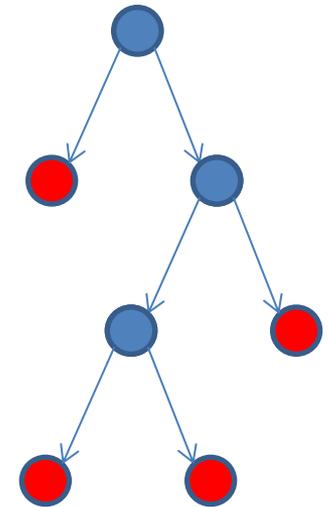


 Подмножества  
(подзадачи)

 терминальные  
(отсеянные) вершины

# Базовый алгоритм ветвей и границ

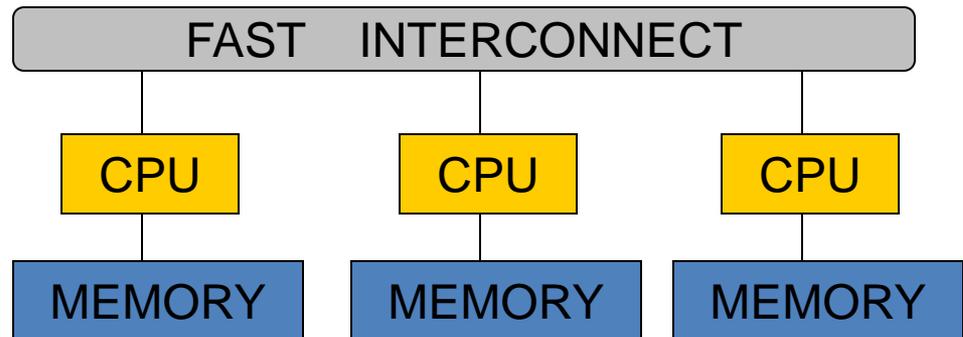
1. Поместить в список подмножеств исходное допустимое множество  $X$
2. Выбрать и удалить из списка некоторое подмножество  $G$ .
3. Произвести действия, направленные на вычисление верхних и нижних оценок целевой функции, сокращение и разбиение подмножества  $G$ . Полученные новые подмножества (если таковые получены) добавить в список.
4. Если список пуст, то закончить алгоритм, в противном случае перейти к шагу 2.



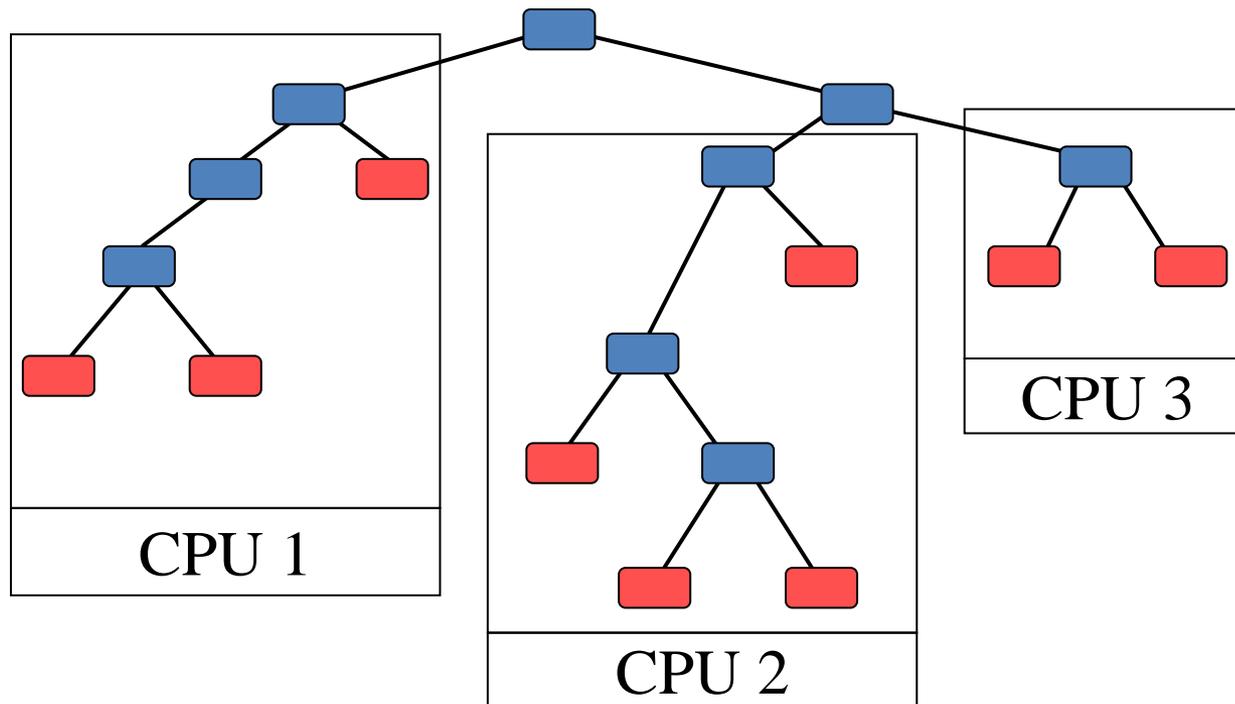
# Системы с распределенной памятью



Супер-компьютеры с  
распределенной и гибридной  
памятью



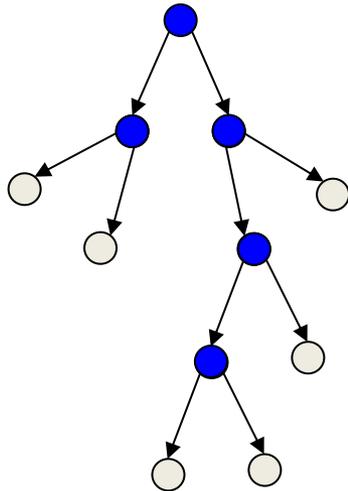
# Параллельная рализация метода ветвей и границ



**ОСНОВА ПОДХОДА:** РАЗЛИЧНЫЕ ВЕТВИ МОГУТ  
ОБРАБАТЫВАТЬСЯ НЕЗАВИСИМО

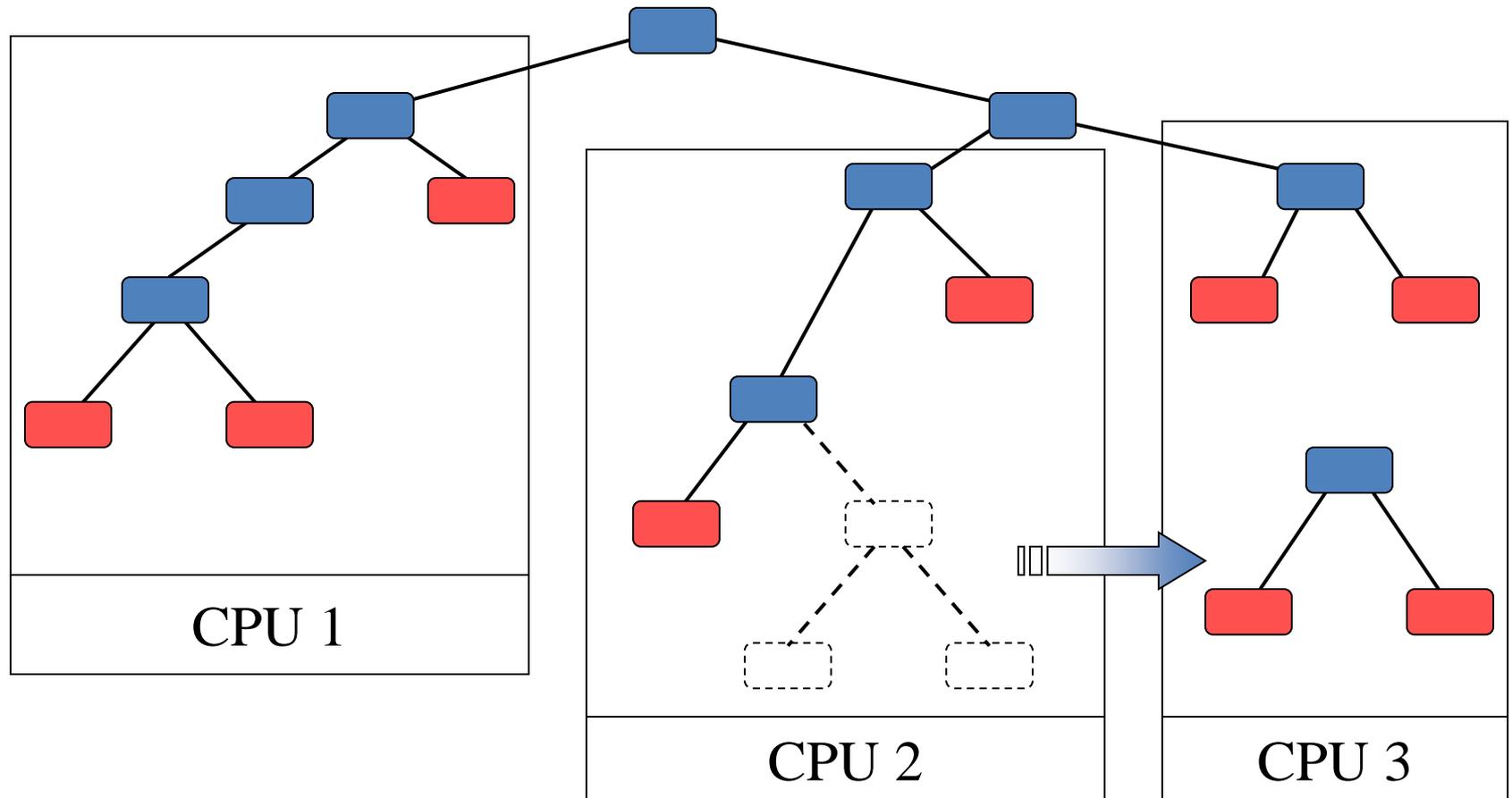
# Проблемы распараллеливания МВГ

ИНФОРМАЦИОННЫЙ  
ГРАФ АЛГОРИТМА

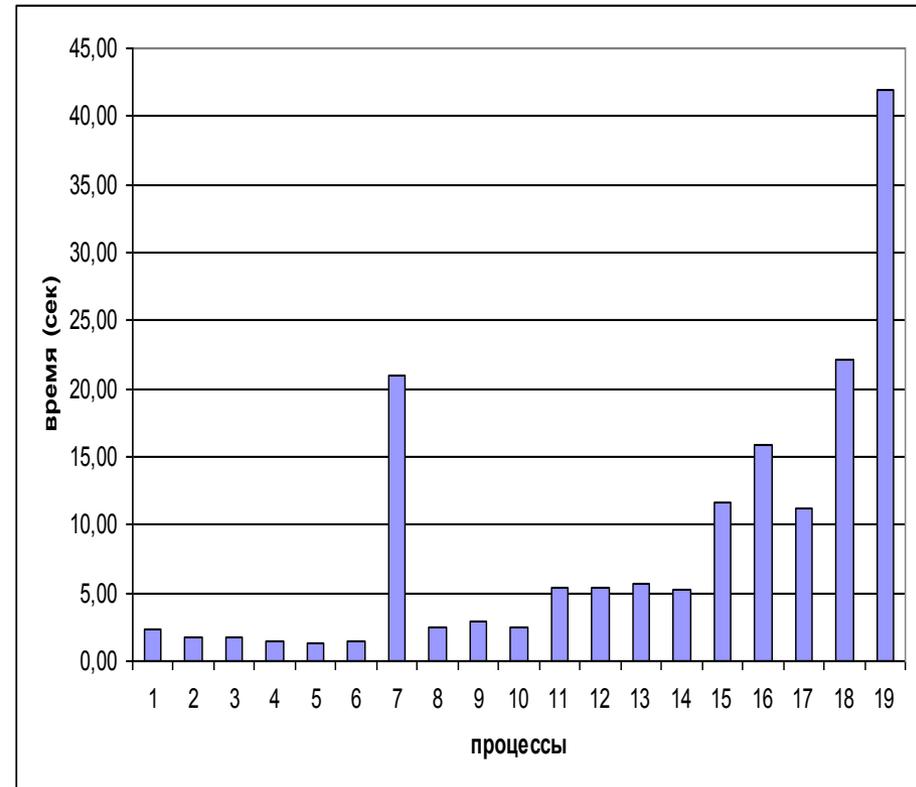
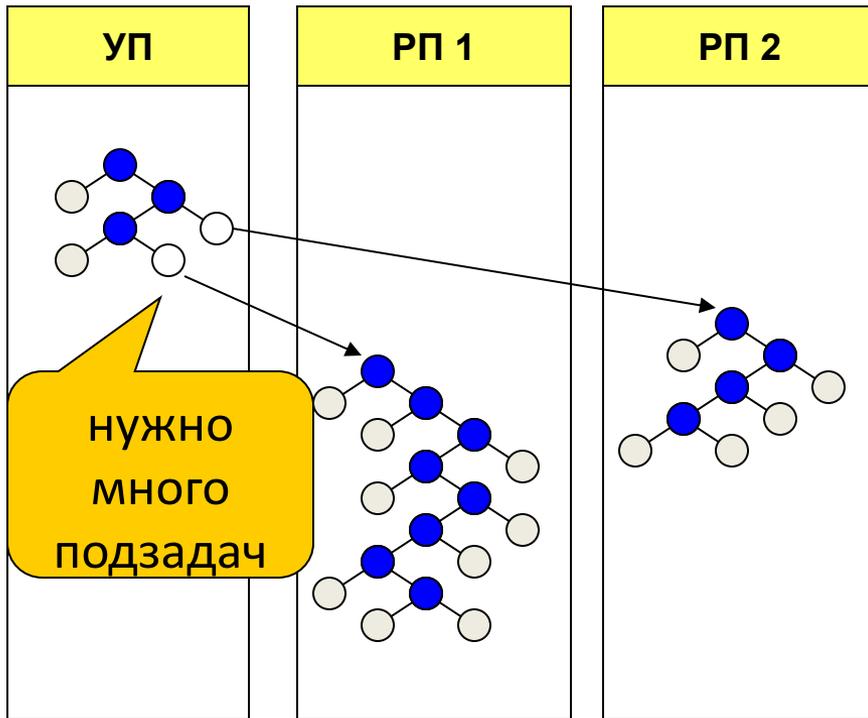


- дерево ветвления не является сбалансированным;
- структура дерева ветвления не известна до начала решения задачи и формируется динамически;

# Основная проблема – разбалансировка нагрузки

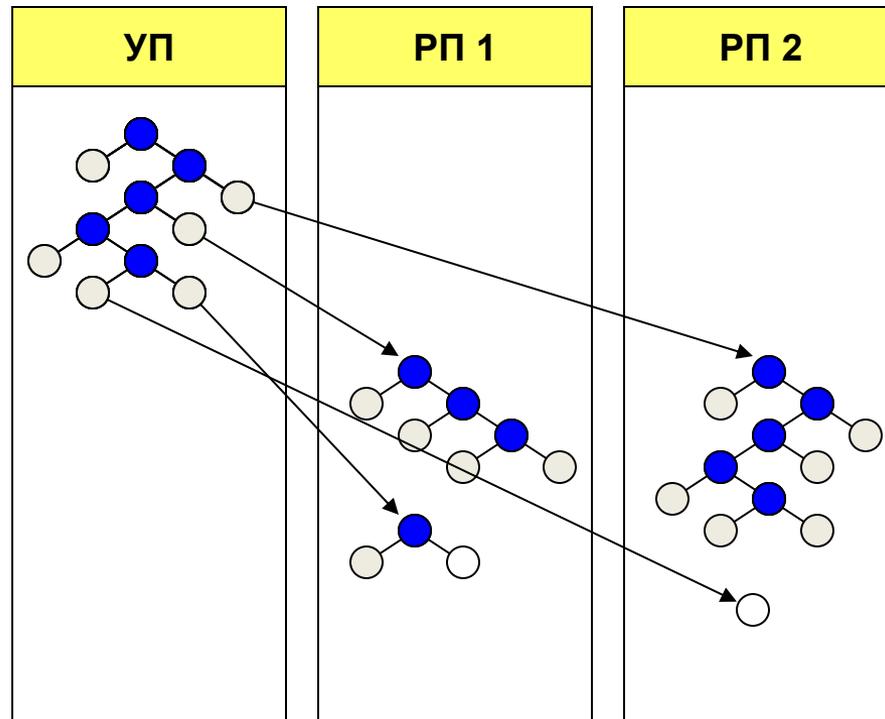


# Разбалансировка нагрузки

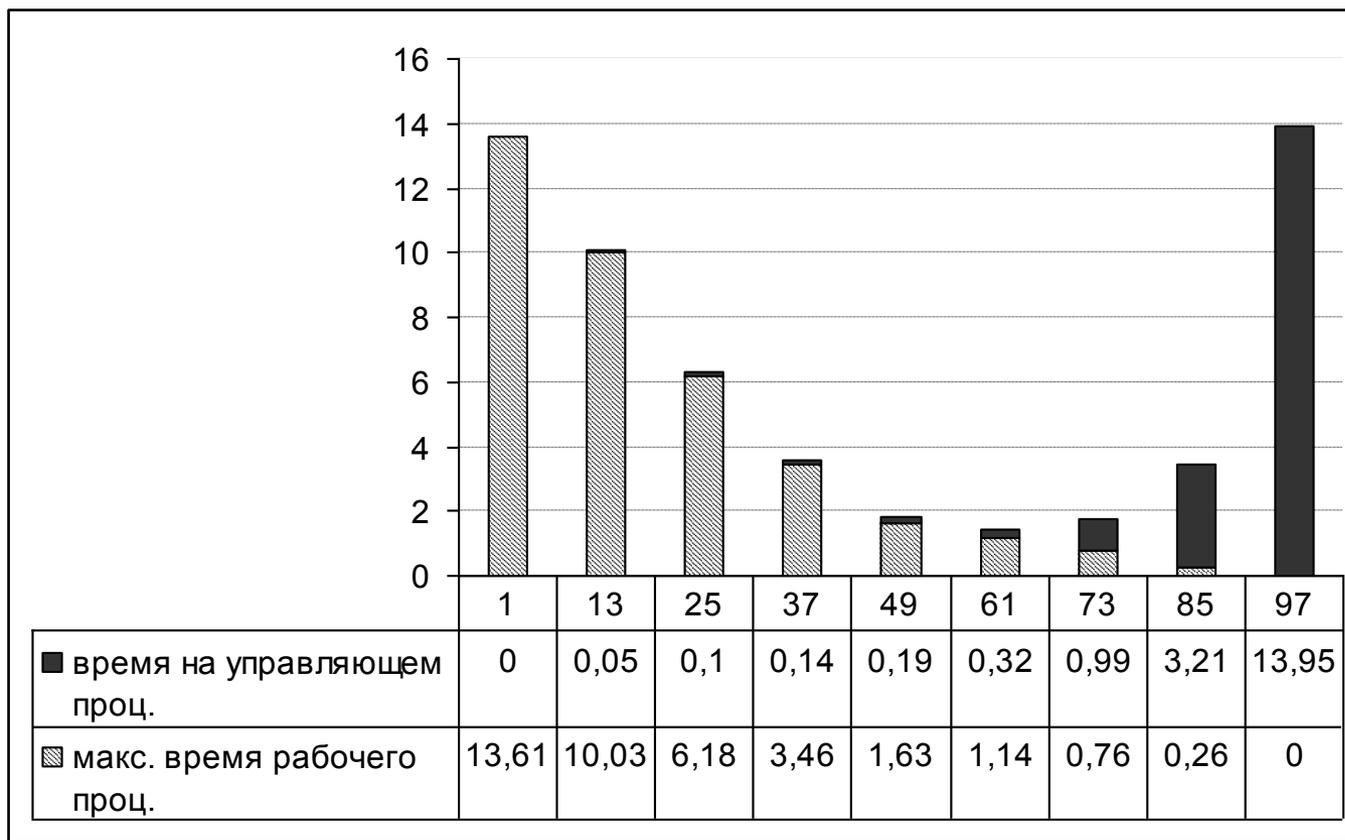


# List Scheduling

У.П. порождает  $N$  подзадач,  
затем  
рассылает их  
рабочим  
процессам по  
мере их  
освобождения



# Распределение времени между У.П. И Р.П. (от *N*)



# Принципиальные ограничения

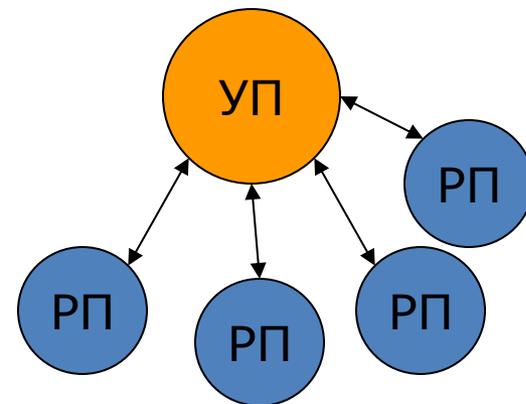
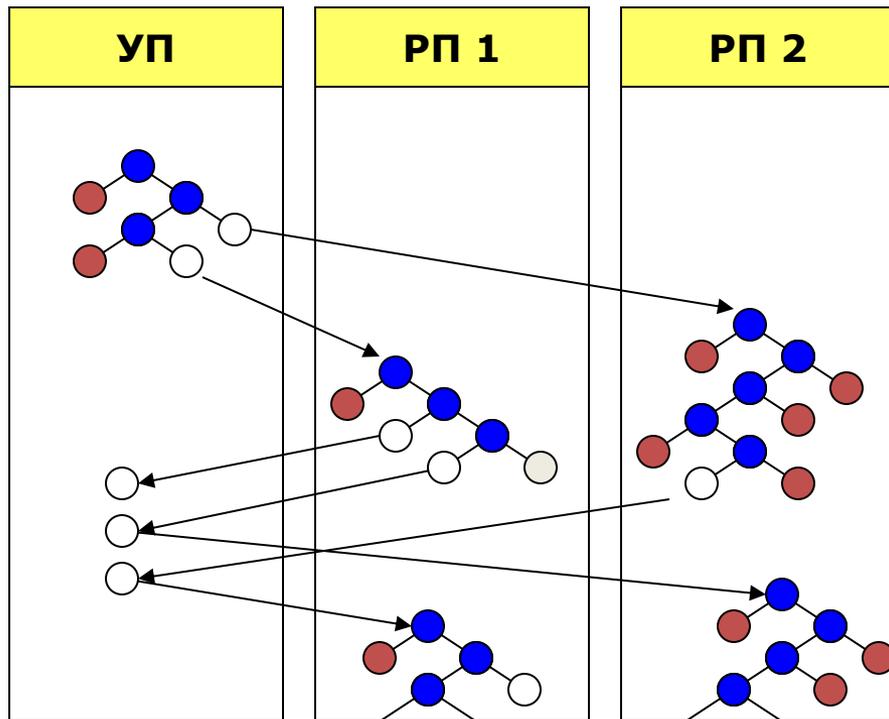
$$L \geq 2\sqrt{S + 2} - 3,$$

$$Sp \leq \frac{S}{2\sqrt{S + 2} - 3} \approx \frac{\sqrt{S}}{2}.$$

*S* – сложность последовательного алгоритма

*L* – сложность параллельного фронтального алгоритма

# Динамическая балансировка нагрузки



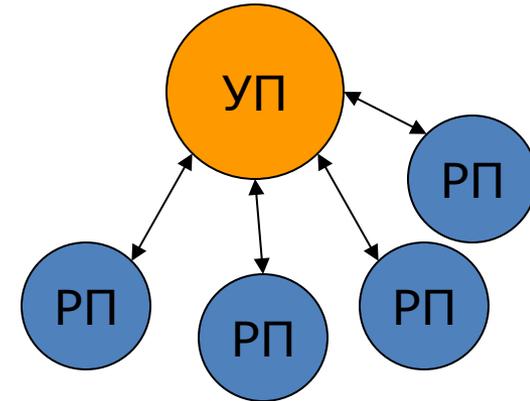
# АДАПТИВНАЯ БАЛАНСИРОВКА НАГРУЗКИ В VNB-SOLVER

**УП** – управляющий процесс, **РП** – рабочий процесс;

$T_b$  – пороговое значение числа ветвлений на рабочем процессе;

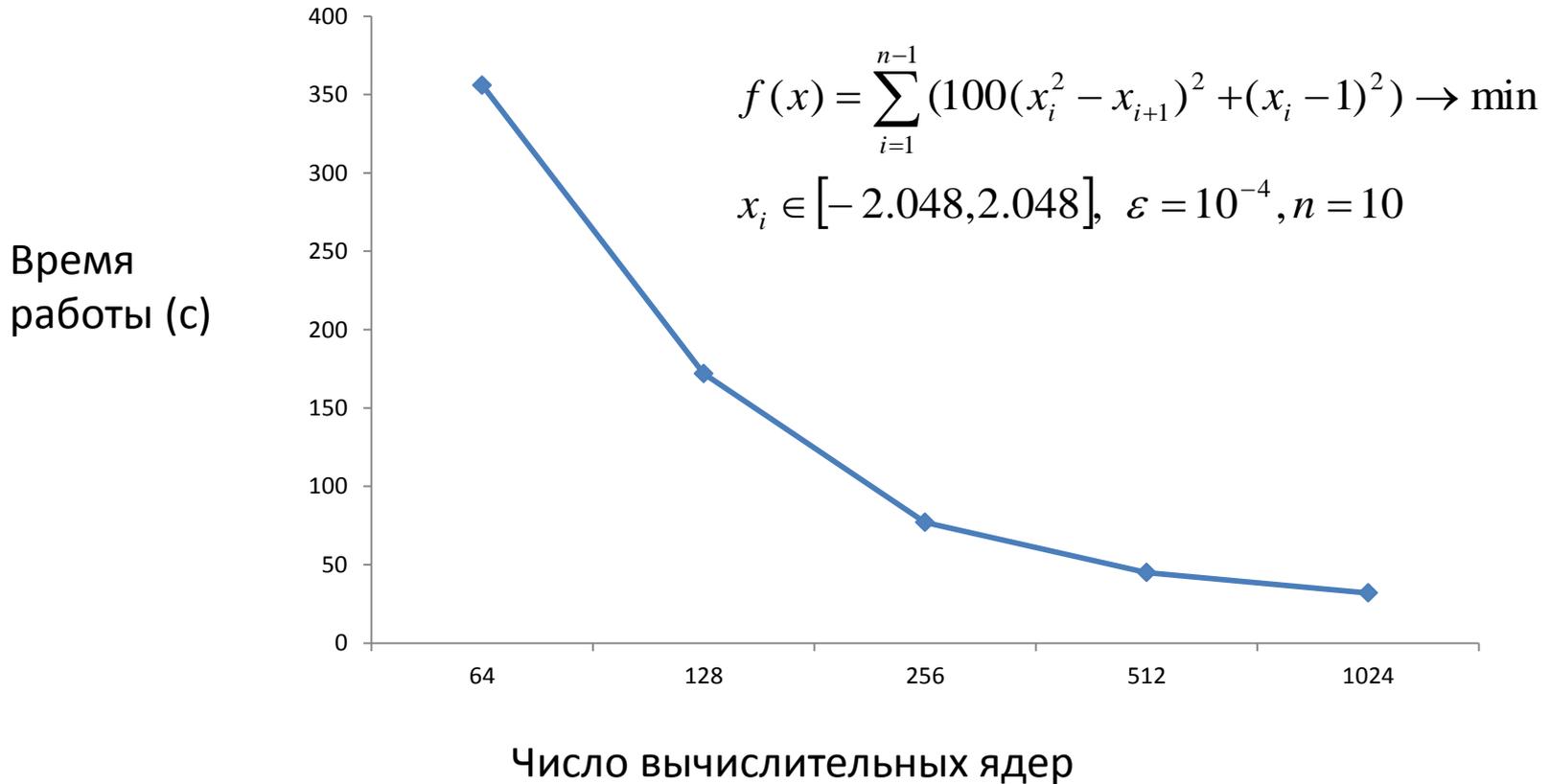
$T_s$  – пороговое значение числа пересылаемых вершин на управляющий процесс;

$U_m(L_m)$  – максимальное (минимальное) число вершин на управляющем процессе



- РП выполняет  $T_b$  ветвлений, после чего посылает УП (не более)  $T_s$  вершин.
- РП процесс, завершивший обработку назначенной вершины, посылает запрос УП. В ответ УП посылает рабочему процессу одну вершину. РП получает ее и начинает выполнять итерации МНП.
- Если на УП скапливается более  $U_m$  вершин, УП посылает сообщение всем РП чтобы они прекратили посылку вершин.
- Если на УП меньше  $L_m$  вершин, то УП посылает сообщение всем РП, чтобы они возобновили посылку вершин.

# Результаты расчетов



# Пример сложной задачи о ранце (Финкельштейн, Венгерова 1974)

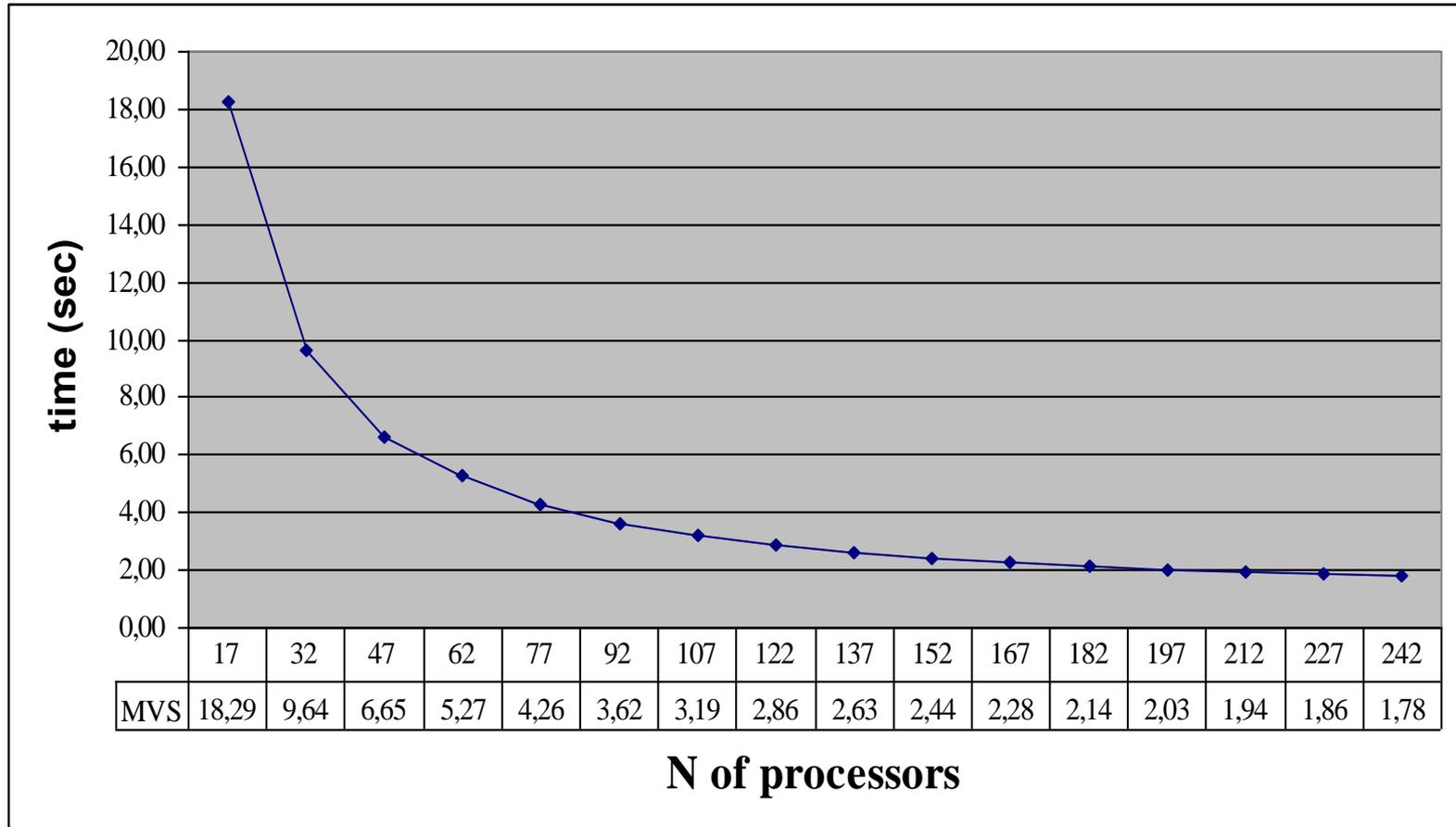
Для задаче о ранце следующего вида

$$\sum_{i=1}^n 2x_i \rightarrow \max$$
$$\sum_{i=1}^n 2x_i \leq 2 \left[ \frac{n}{2} \right] + 1$$

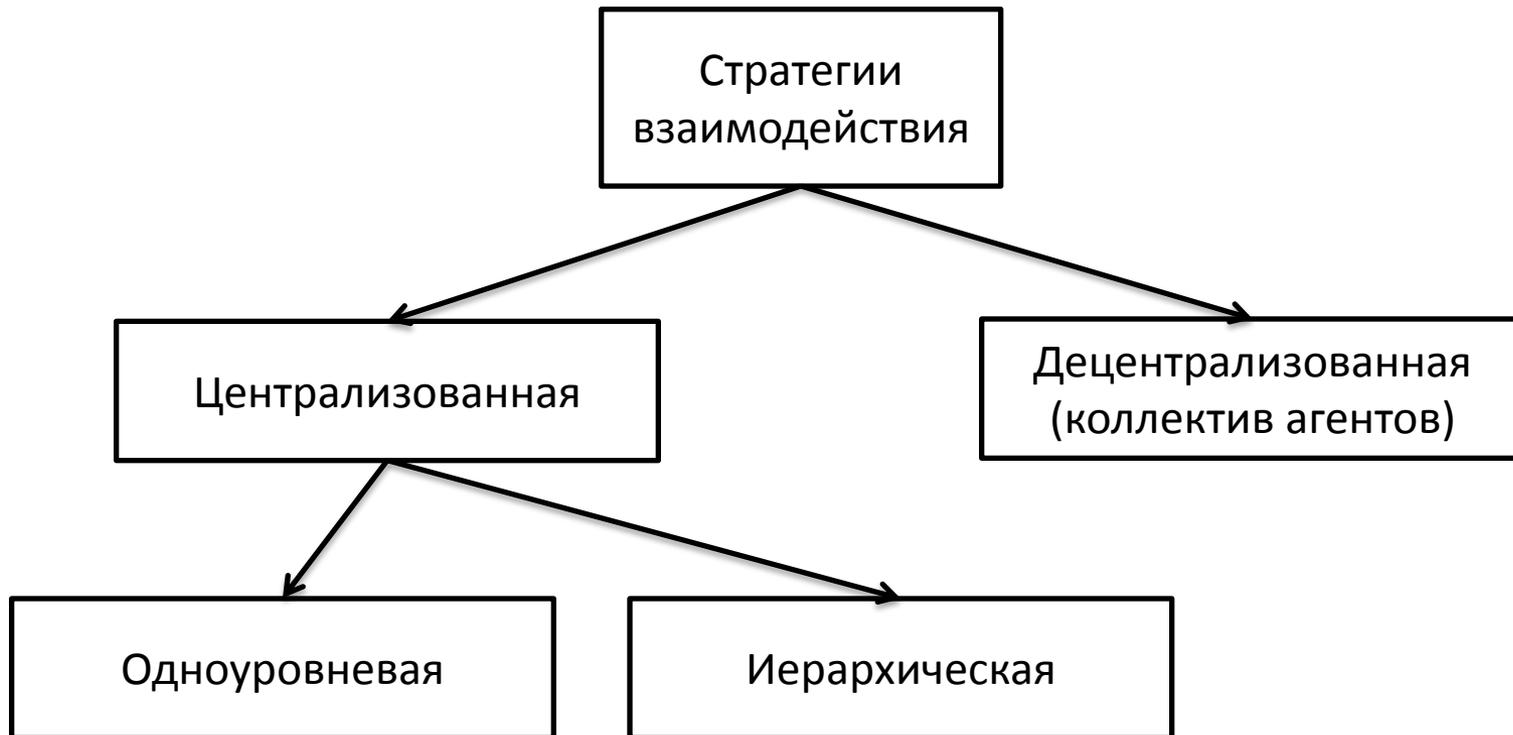
метод ветвей и границ требует асимптотически

$$\frac{2^n}{\sqrt{n}} \quad \text{ветвлений}$$

# Результаты эксперимента (MVS 15000 bm)



# Организация взаимодействия процессов



# Типы балансировки

- Push strategy
- Pull strategy
- Hybrid strategy

*Gendron B., Crainic T. G. Parallel branch-and-branch algorithms: Survey and synthesis //Operations research. – 1994. – Т. 42. – №. 6. – С. 1042-1066.*

# Метод ветвей и границ

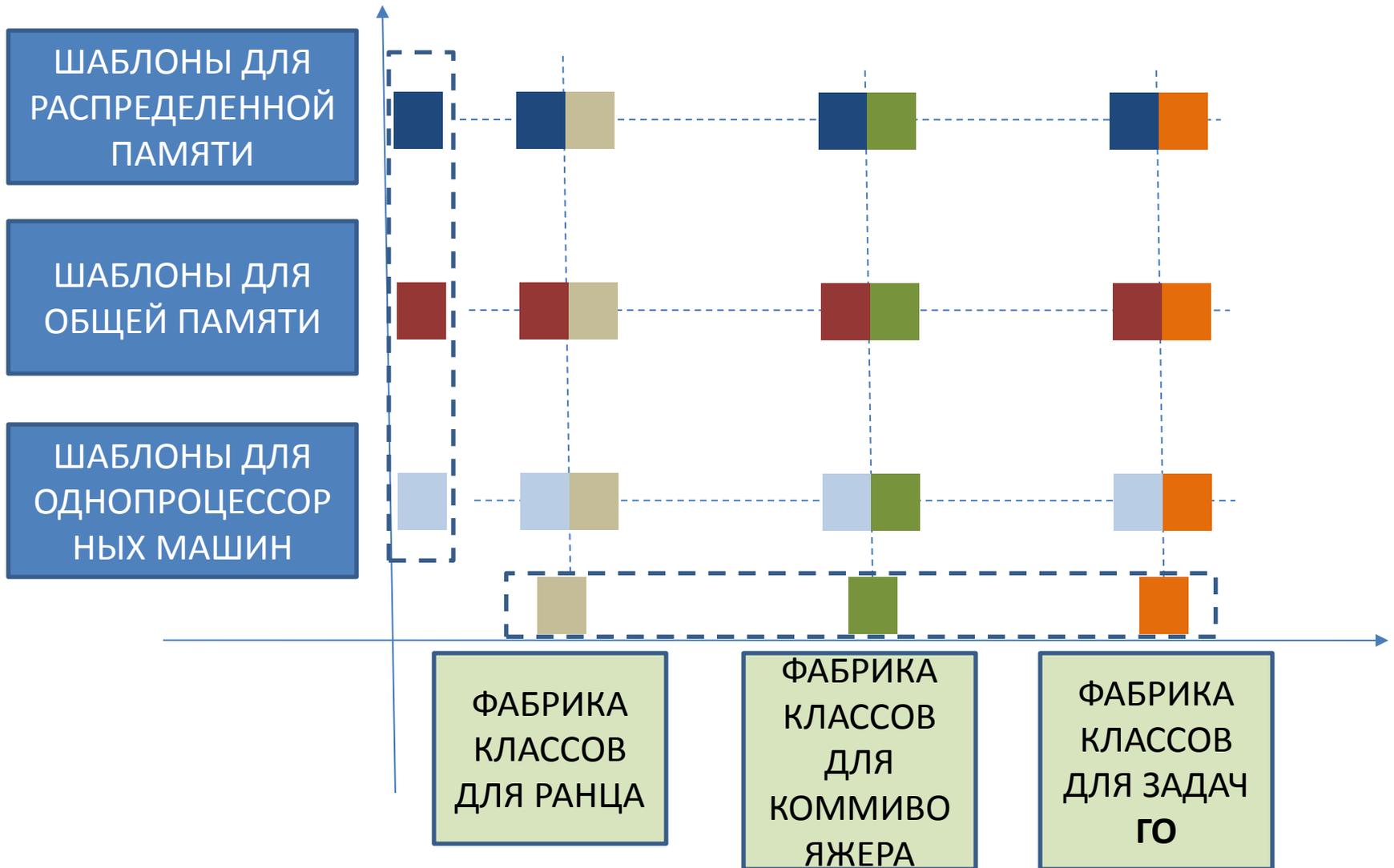
## ПРОБЛЕМНО-НЕЗАВИСИМЫЕ

- Организация хранения подмножеств и допустимых решений
- Общая схема вычислений

## ПРОБЛЕМНО-ЗАВИСИМЫЕ

- Способ ветвления
- Правила отсева

# Программный комплекс VNB-Solver



# Программный комплекс BNB-Solver

СОЛВЕРЫ

МВГ ДЛЯ ЗАДАЧИ  
О РАНЦЕ

МЕТОД  
НЕРВАНОМЕРНЫХ  
ПОКРЫТИЙ ДЛЯ ЗАДАЧ  
НЛП

МЕТОД НЕРАВНОМЕРНЫХ ПОКРЫТИЯ  
ДЛЯ ЗАДАЧ МКО

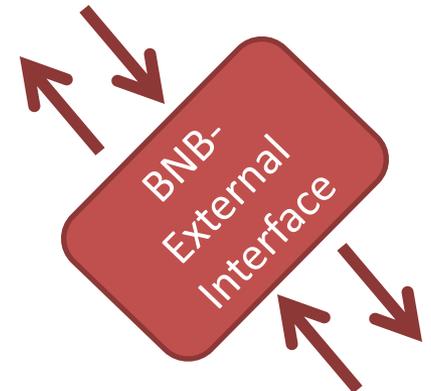
ШАБЛОН  
ДЛЯ ОБЩЕЙ  
ПАМЯТИ

ШАБЛОН  
ДЛЯ  
РАСПРЕДЕЛЕННОЙ  
ПАМЯТИ

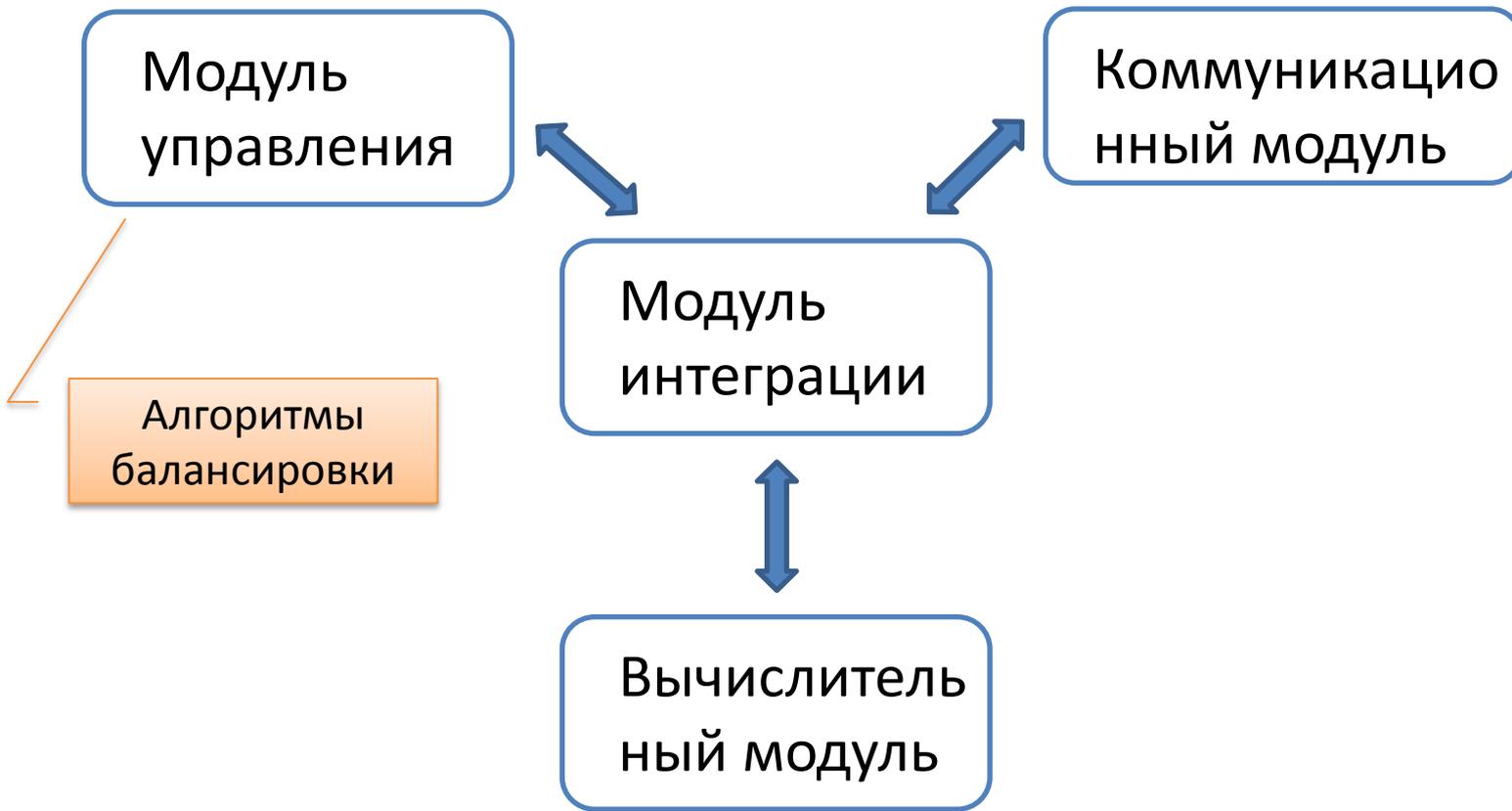
ШАБЛОН  
ДЛЯ  
ПОСЛЕД.  
АРХИТЕКТУР

БАЗОВЫЕ  
ВЫЧИСЛИТЕЛЬНЫЕ  
ФУНКЦИИ

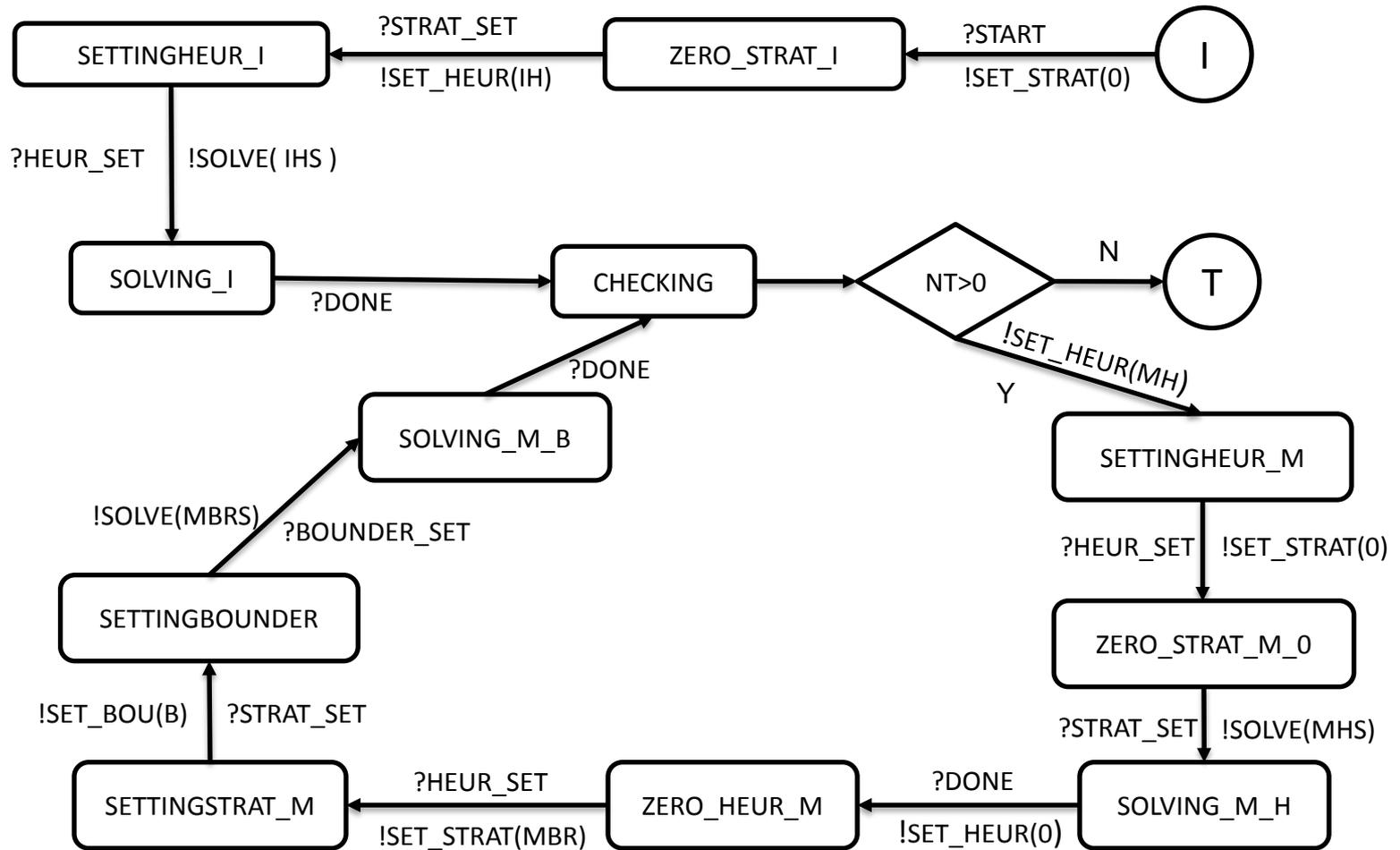
БАЗОВЫЕ  
КОММУНИКАЦИОННЫЕ  
ФУНКЦИИ



# СХЕМА ПРОГРАММНОГО КОМПЛЕКСА



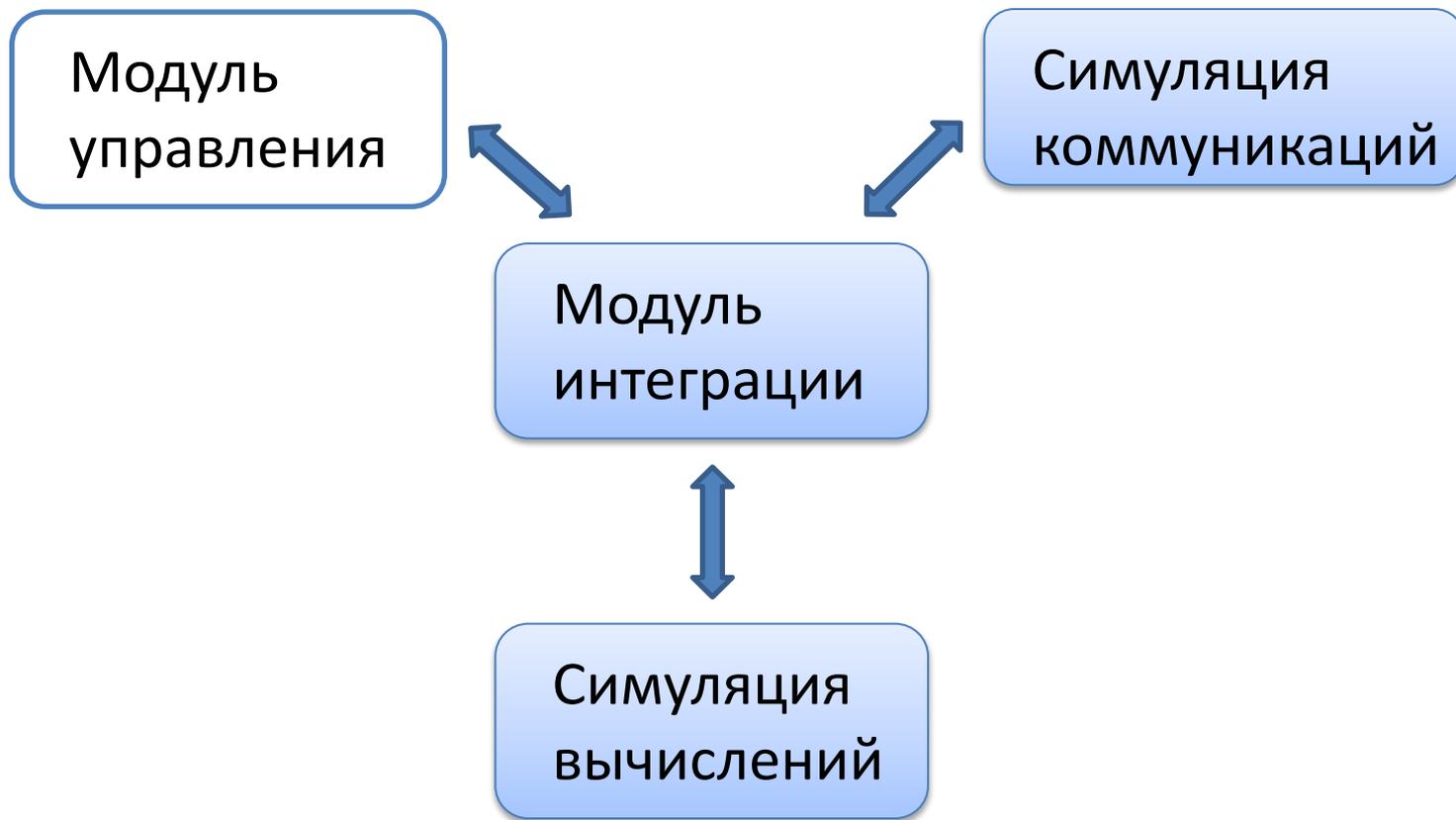
# Автомат балансировщика нагрузки (управляющий процесс)



# BNB-Simulator

- Имитирует метод ветвей и границ ветвящимся процессом
- Имитирует передачу данных
- Использует балансировщики BNB-Solver без адаптации

# BNB-Simulator



# Запуск симулятора

BnBVisualizer

File Help

Compute trace Plot visualization Processors table visualization Data exchange visualization Statistics

**Time cost**

solve:  store:

load:  overhead:

**Parcel size**

command:  record:  subproblem:

**Communicator and resolver**

latency:  bandwidth:  max task level:

**Simulator parameters and input**

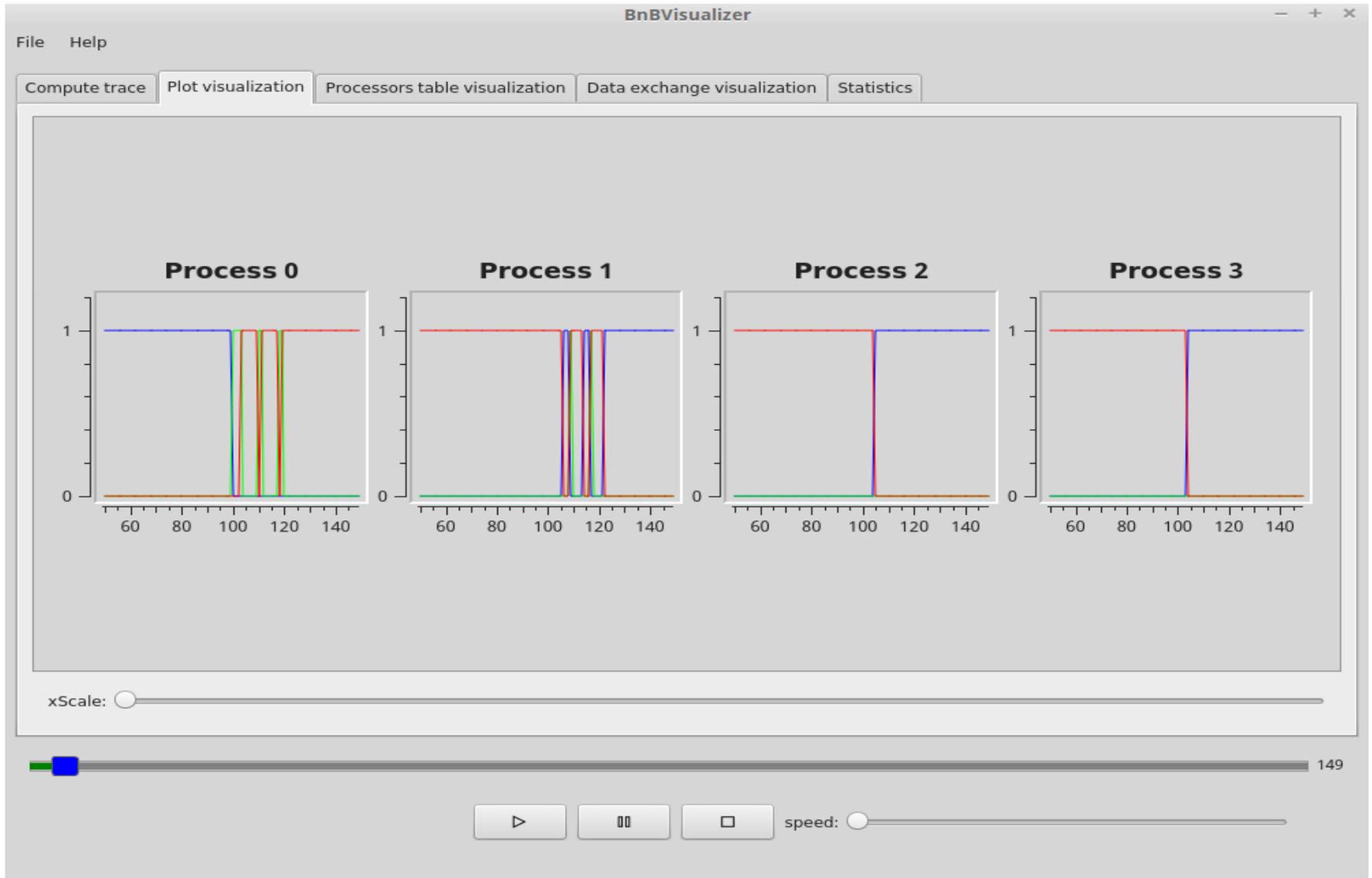
number of process:  input:

Save as:  .trc

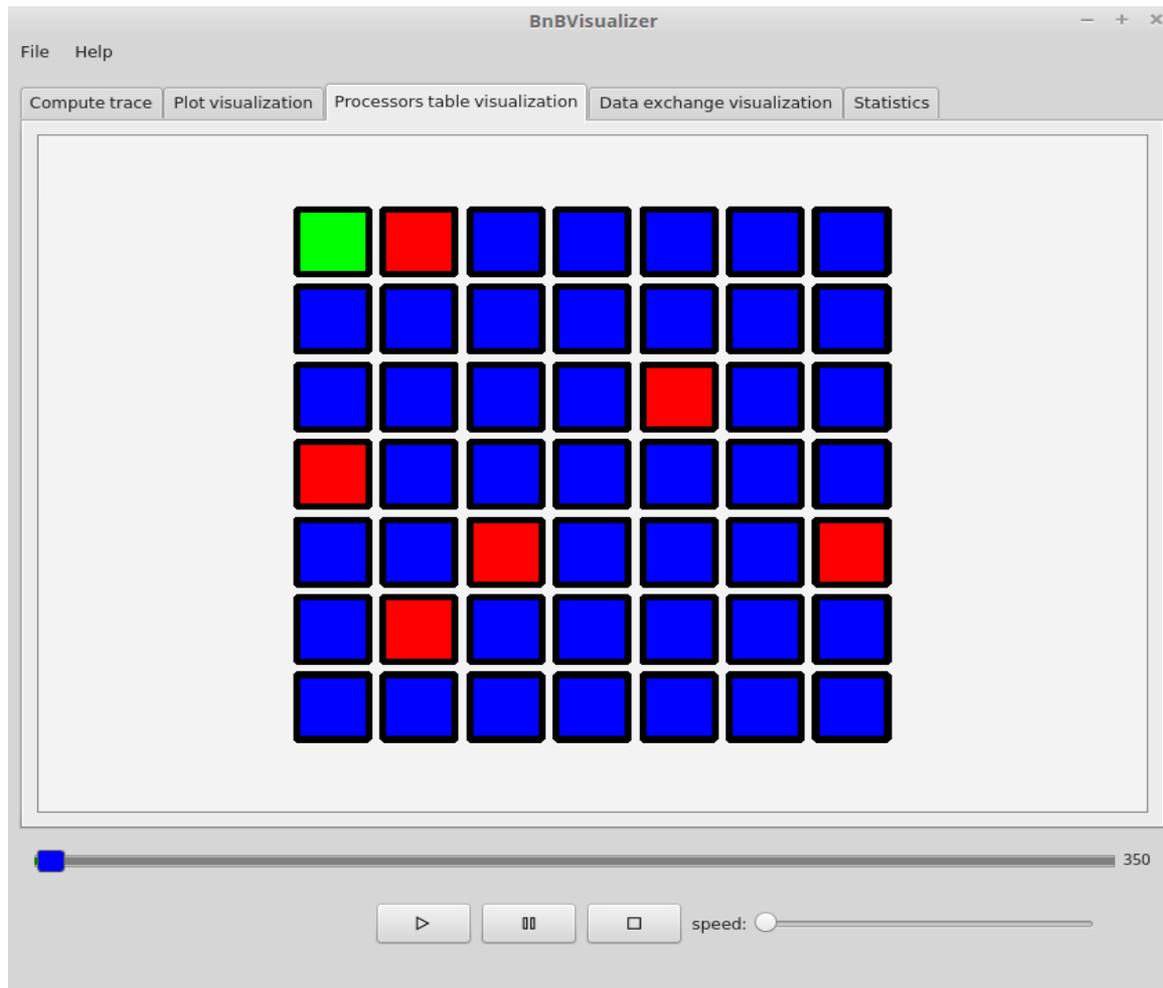
speed:

0

# ВИЗУАЛИЗАЦИЯ ГРАФИКОВ

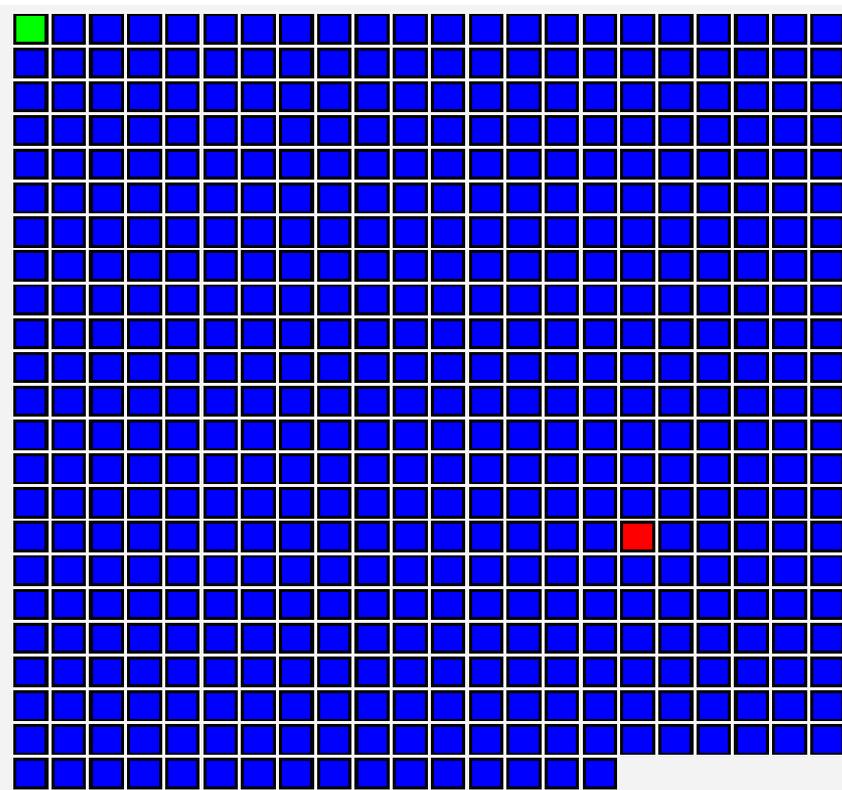
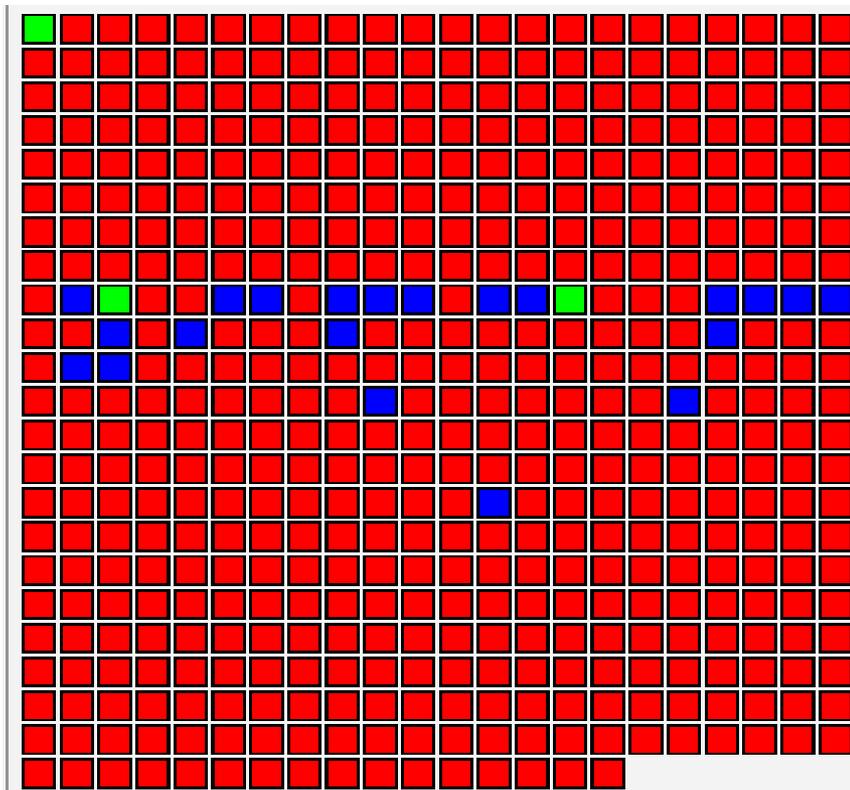


# ТАБЛИЦА ПРОЦЕССОРОВ

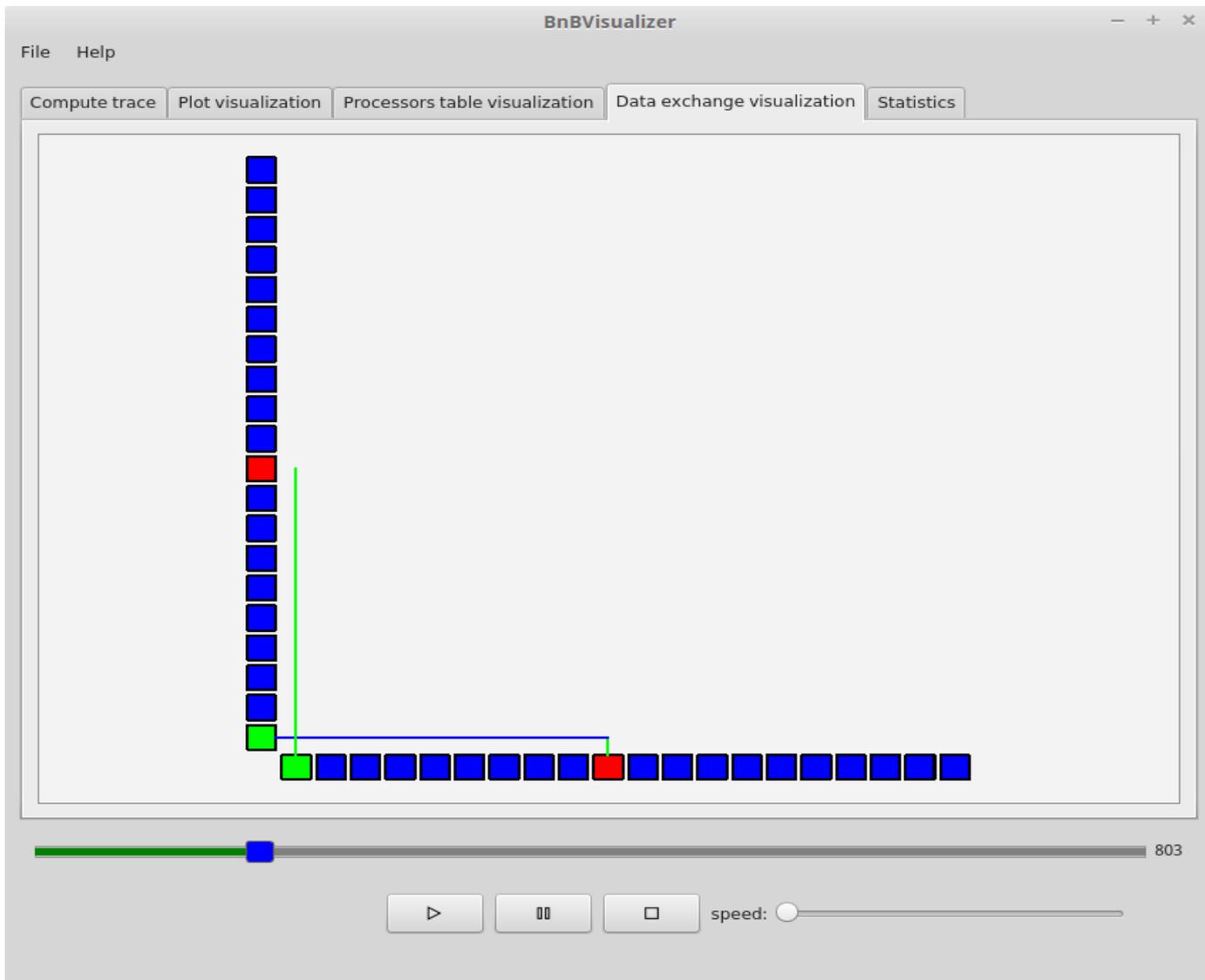


-  состояние ожидания
-  состояние счета
-  состояние отправки данных

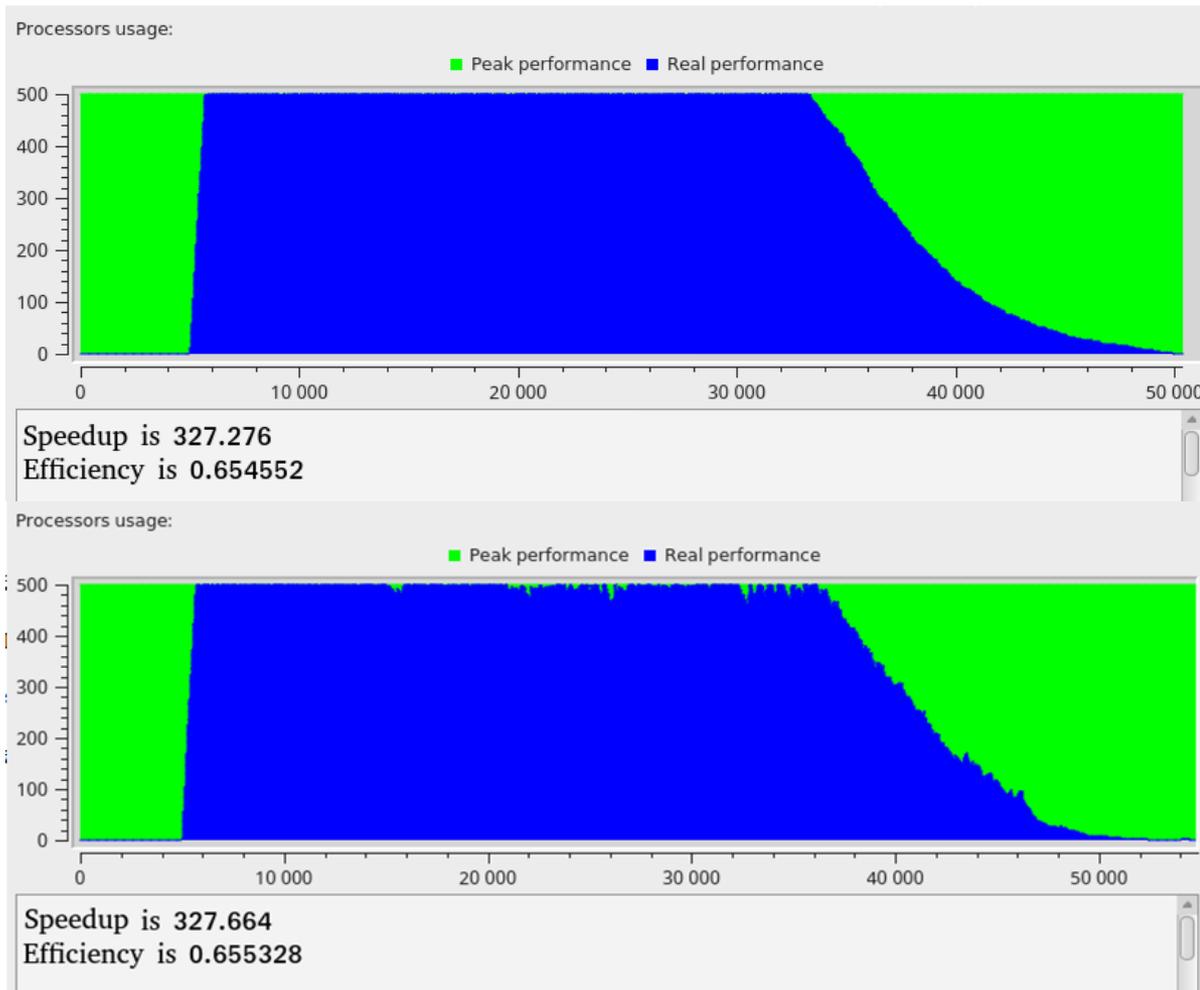
# ТАБЛИЦЫ ПРОЦЕССОРОВ



# ВИЗУАЛИЗАЦИЯ ОБМЕНА ДАННЫМИ



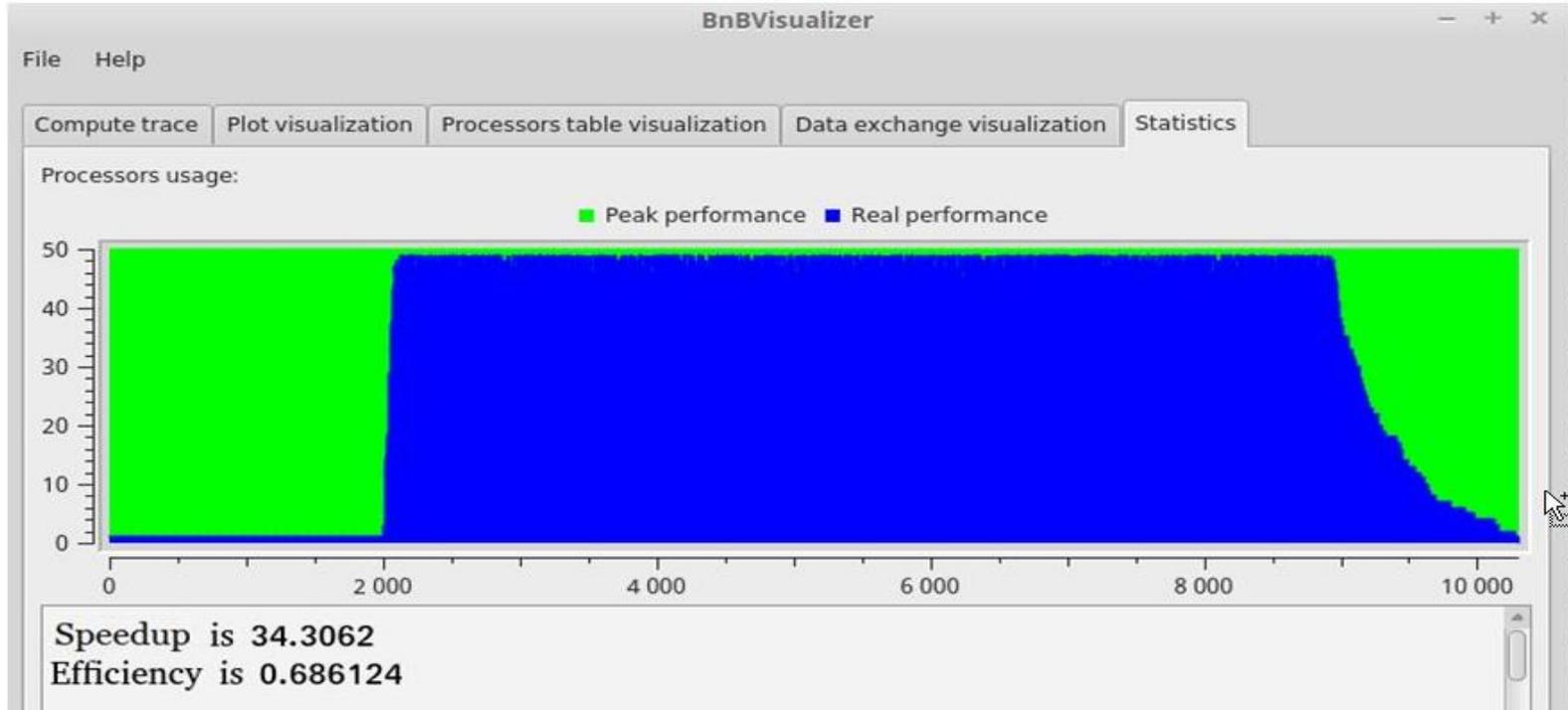
# СРАВНЕНИЕ СТАТИСТИКИ ПО РЕЗУЛЬТАТАМ ЭКСПЕРИМЕНТАЛЬНОГО ИССЛЕДОВАНИЯ СТАНДАРТНОГО И АДАПТИВНОГО АЛГОРИТМОВ БАЛАНСИРОВКИ НАГРУЗКИ НА 500 ПРОЦЕССОРАХ



$$S = \frac{t_{\text{пар}}}{t_{\text{пол}}};$$

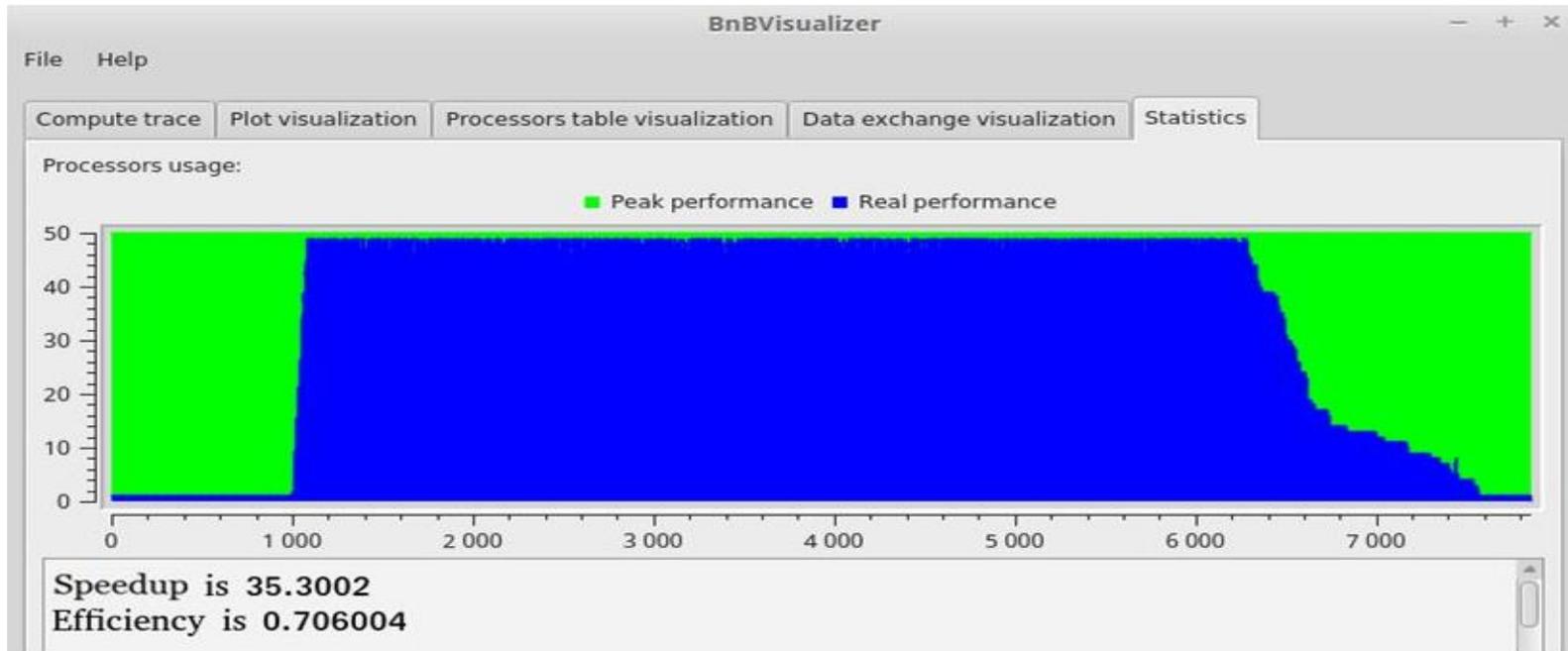
$$E = \frac{S}{n}.$$

# АЛГОРИТМ ПАРАЛЛЕЛЬНОЙ БАЛАНСИРОВКИ НАГРУЗКИ



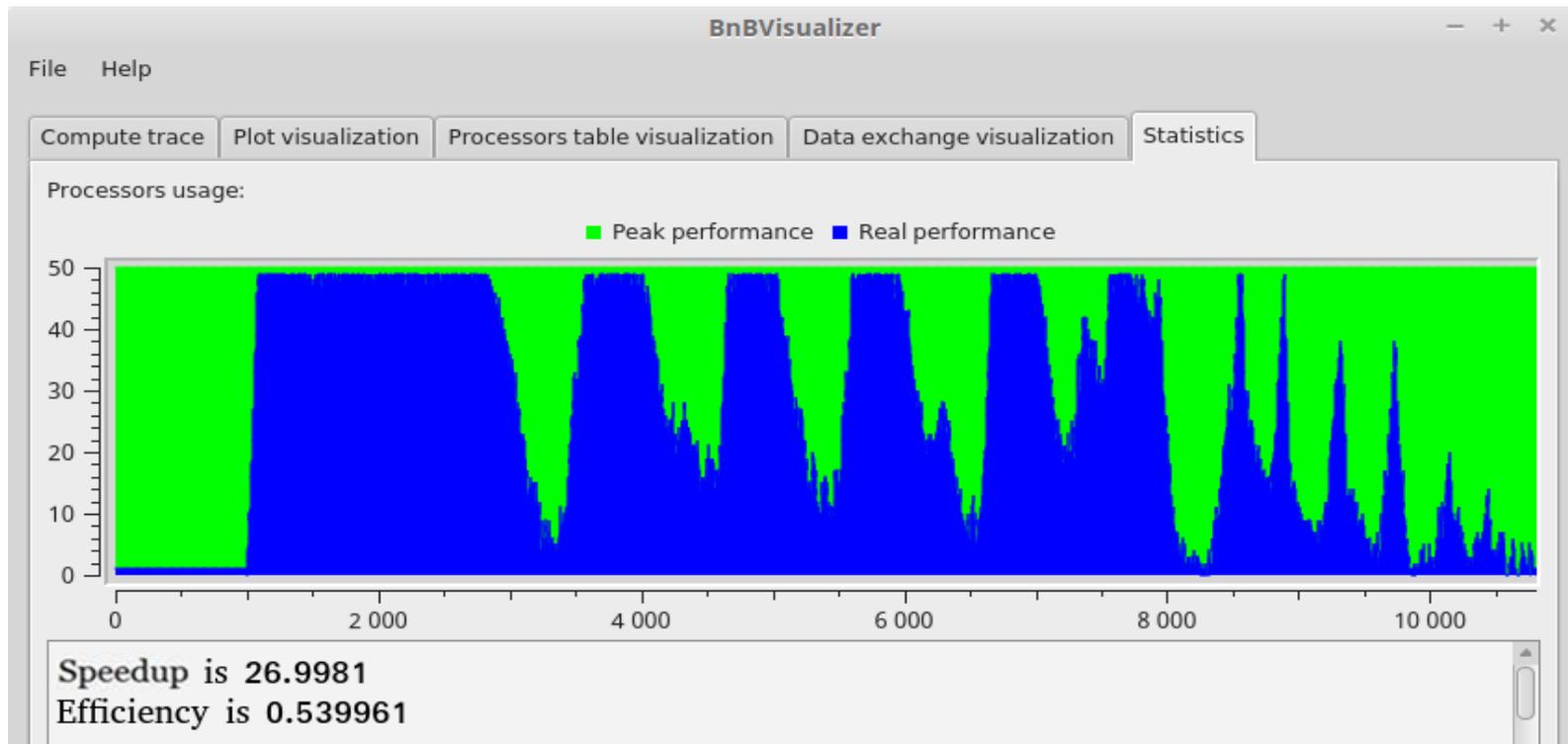
- Время выполнения последовательных действий достаточно велико
- Плохая балансировка дерева подзадач на этапе завершения работы

# АЛГОРИТМ АДАПТИВНОЙ БАЛАНСИРОВКИ НАГРУЗКИ С ФИКСИРОВАННЫМИ ПАРАМЕТРАМИ



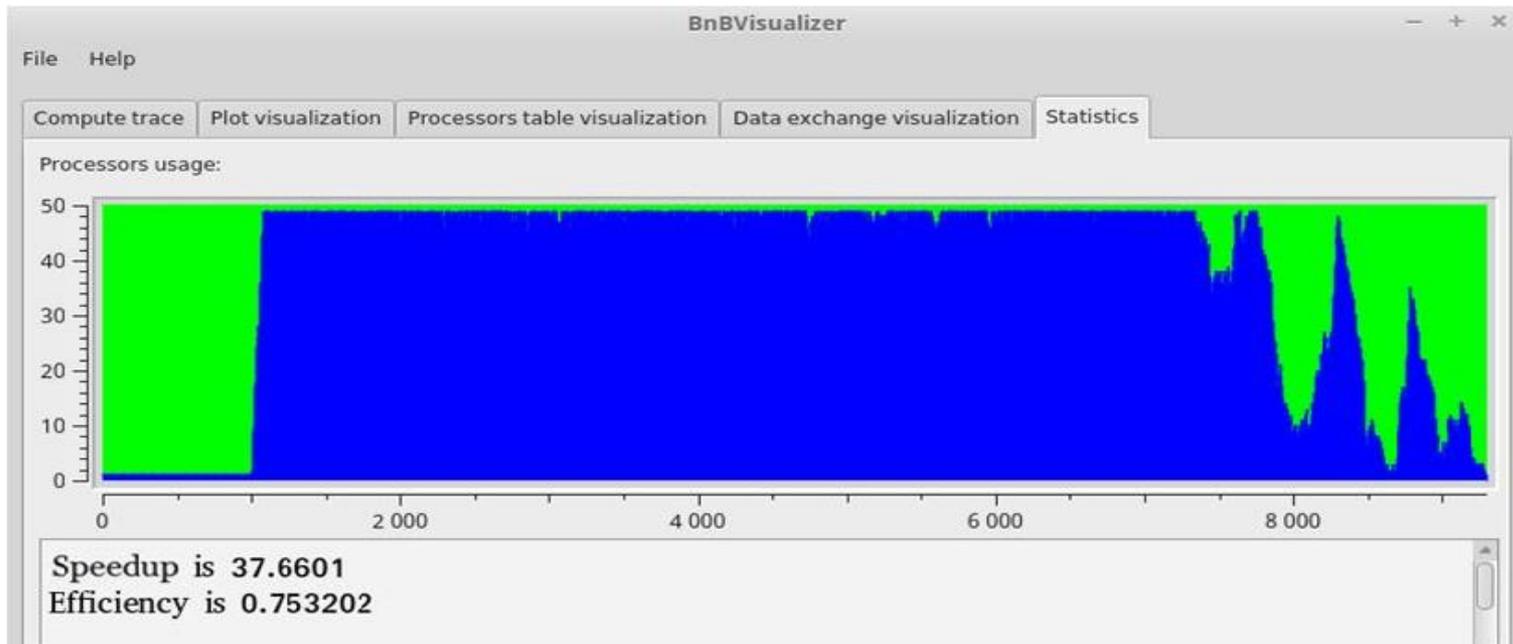
- Время выполнения последовательных действий сокращено, вместе с тем возросла эффективность
- Плохая балансировка дерева подзадач на этапе завершения работы всё еще имеет место

# АЛГОРИТМ АДАПТИВНОЙ БАЛАНСРОВКИ НАГРУЗКИ С ИЗМЕНЯЮЩИМИСЯ ПАРАМЕТРАМИ



- Малое число ветвлений на рабочих процессах приводит к увеличению накладных расходов

# АЛГОРИТМ АДАПТИВНОЙ БАЛАНСРОВКИ НАГРУЗКИ С ИЗМЕНЯЮЩИМИСЯ ПАРАМЕТРАМИ



- При полной загрузке рабочих процессов нет смысла часто прерывать их работу
- При малом количестве подзадач на управляющем процессе прерывание работы рабочих процессов происходит чаще

**БЛАГОДАРЮ ЗА ВНИМАНИЕ!**

# Детерминированные методы

- Методы динамического программирования
- Методы отсечений
- Методы ветвей и границ
- Методы ветвей и отсечений
- ...

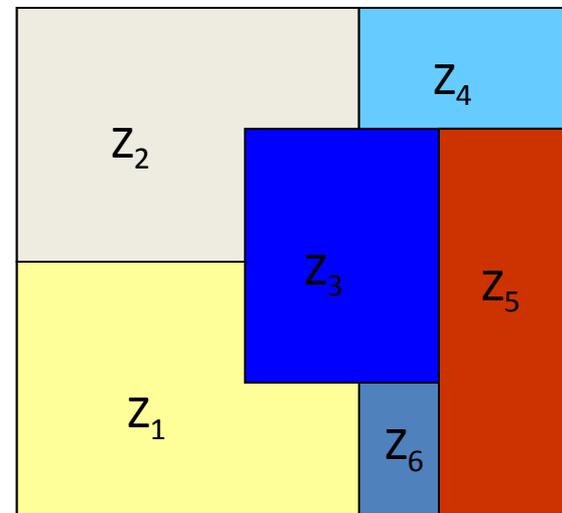
# Схема ветвей и границ

Применяется в качестве алгоритмической основы многих методах для широкого спектра задач оптимизации (с одним и несколькими критериями)

- метод неравномерных покрытий (Ю.Г. Евтушенко)
- метод секущих (Р.Г. Стронгин, В.П. Гергель, Я.Д. Сергеев )
- методы интервального анализа (Hansen, Шарый)
- ....

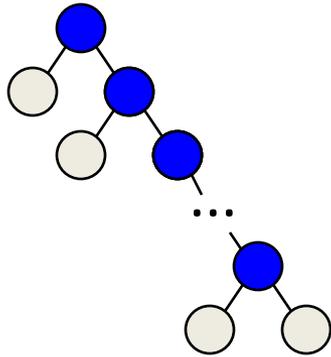
$$f(x) : X \rightarrow R, X \subseteq R^n$$

$$f(x) \rightarrow \min_{x \in X}$$

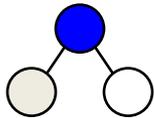


# Пример «плохой» задачи

ДЕРЕВО ВЕТВЛЕНИЯ



НА КАЖДОМ ШАГЕ – РОВНО 1  
ЗАДАЧА-КАНДИДАТ:



**УСКОРЕНИЕ**  $\leq 1$

(для любого алгоритма из класса **РВВ**)

ЗАДАЧА О РАНЦЕ

$$f(x) = \sum_{i=1}^n c_i x_i \rightarrow \max, \quad \sum_{i=1}^n a_i x_i \leq R,$$

$$x_i \in \{0,1\}, \quad i = 1,2,\dots,n$$

$$a_1 = c_1 = 2$$

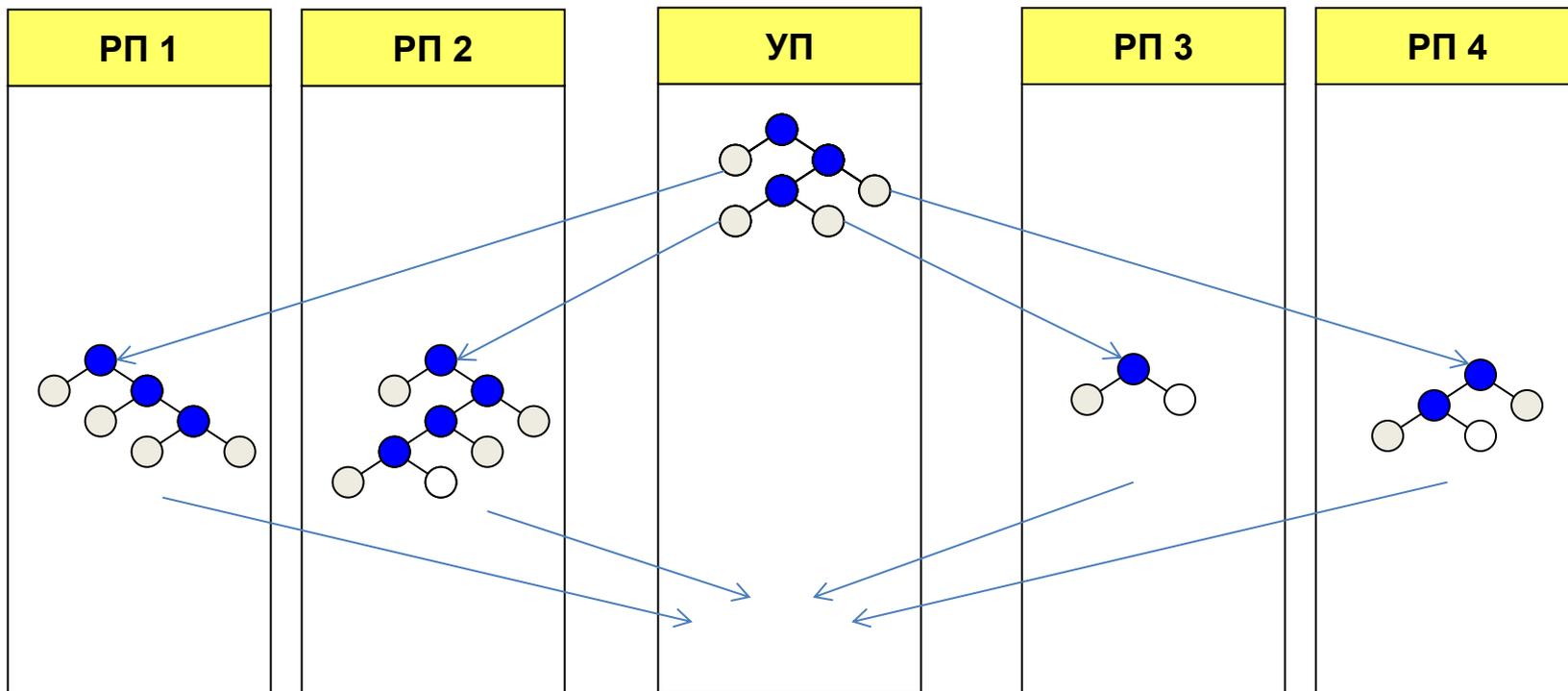
$$a_i = c_i = 2^{i-1} \text{ для } i \in \overline{2,n}$$

$$R = 2^n - 1$$

# Простейший подход

У.П. порождает  $N$  подзадач,

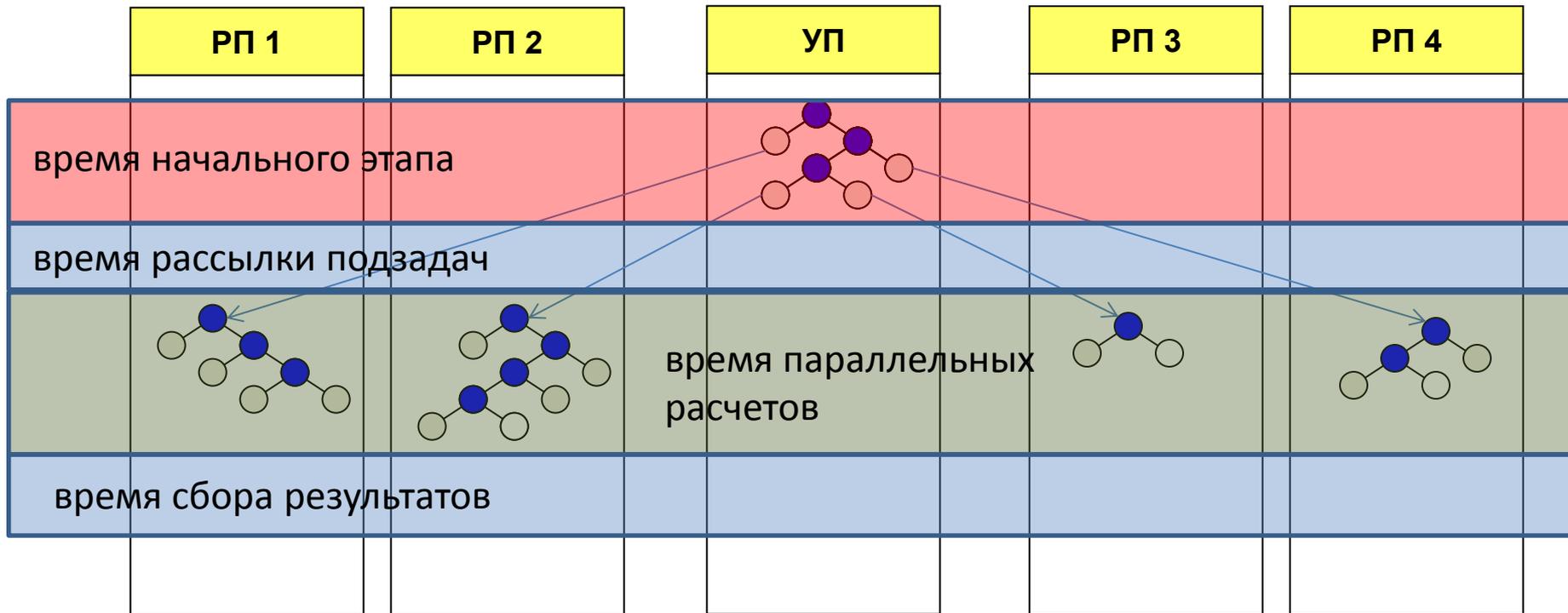
затем рассылает их рабочим процессам по одной на каждую



# Простейший подход

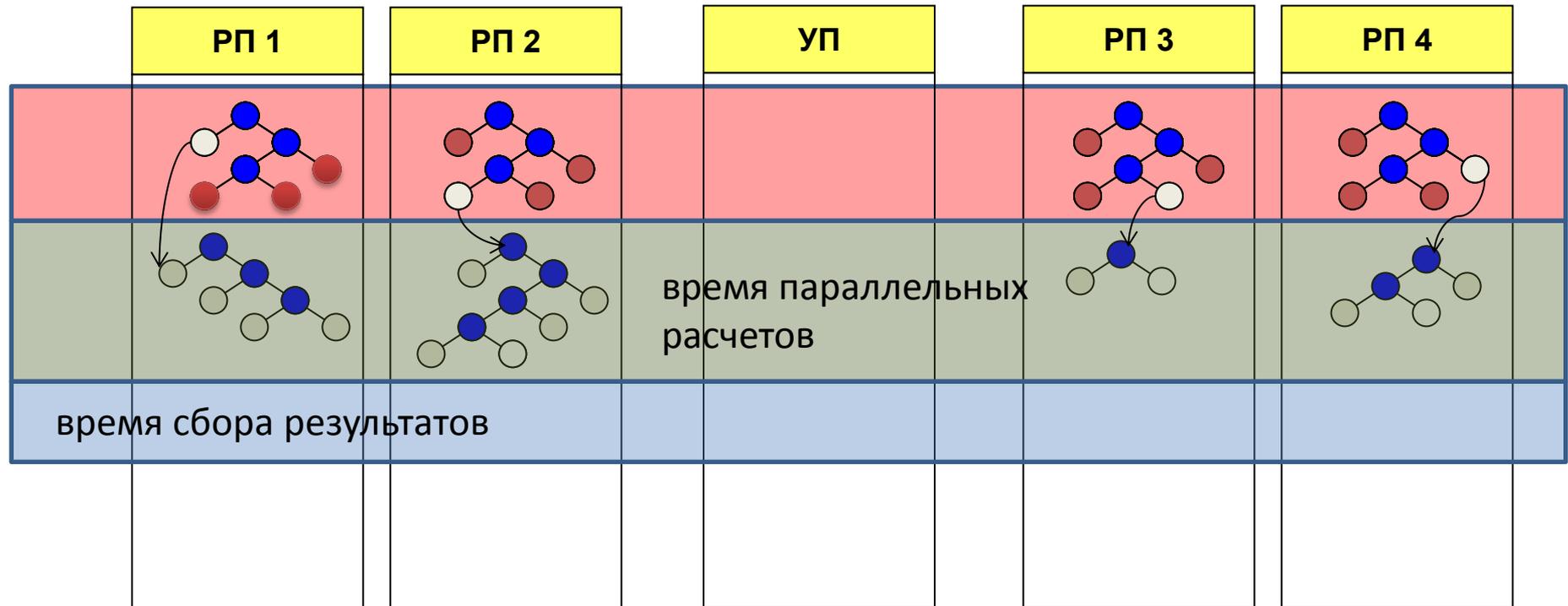
У.П. порождает  $N$  подзадач,

затем рассылает их рабочим процессам по одной на каждую



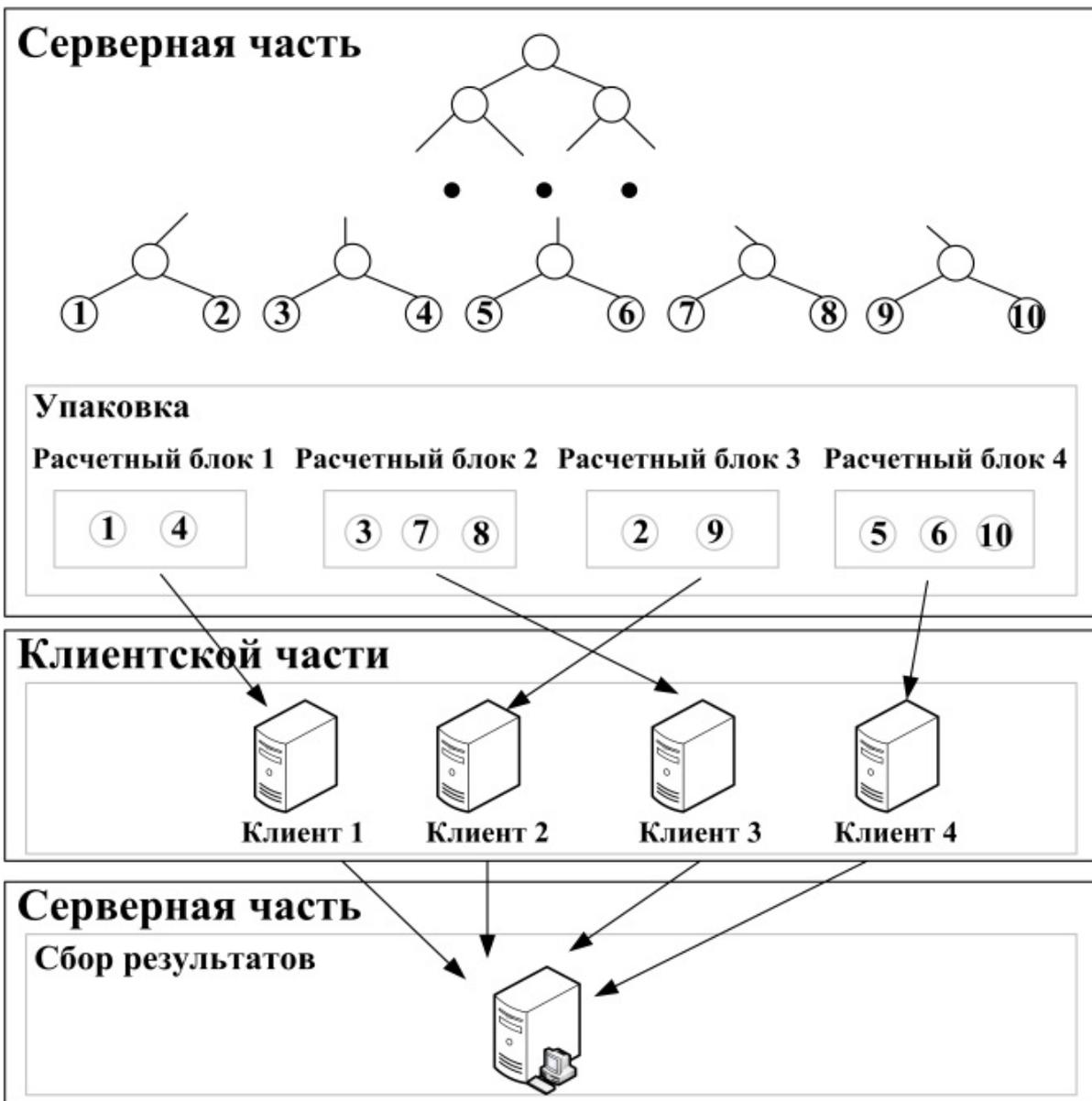
# Сокращение времени на пересылку

Первый этап дублируется на рабочих процессах



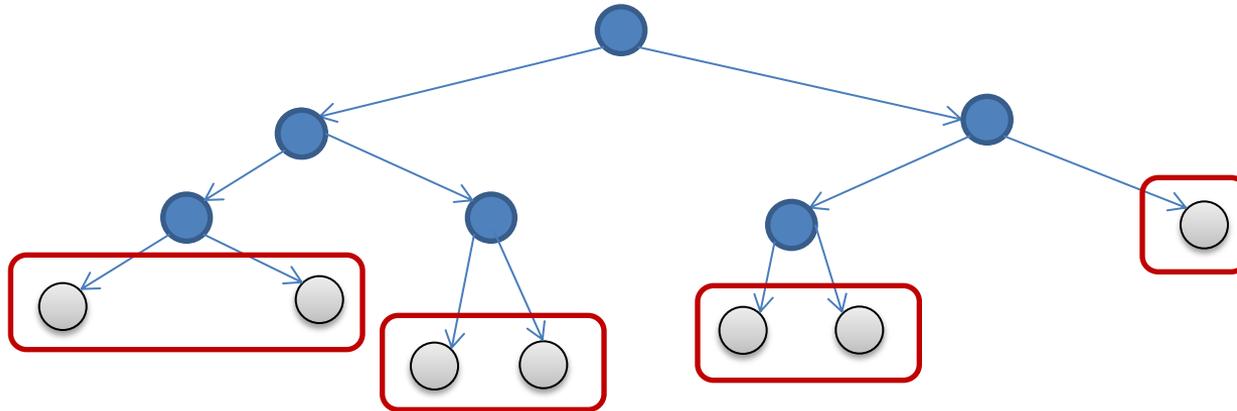
*Fischetti, M., Monaci, M., & Salvagnin, D. (2014). Self-splitting of workload in parallel computation. In Integration of AI and OR Techniques in Constraint Programming (pp. 394-404). Springer International Publishing.*

# Борьба с дисбалансом – агрегация



# Простые способы агрегации

1. «Плотная стратегия» - соседние узлы помещаются в один блок



2. «Случайная стратегия» - расчетные блоки формируются случайным образом

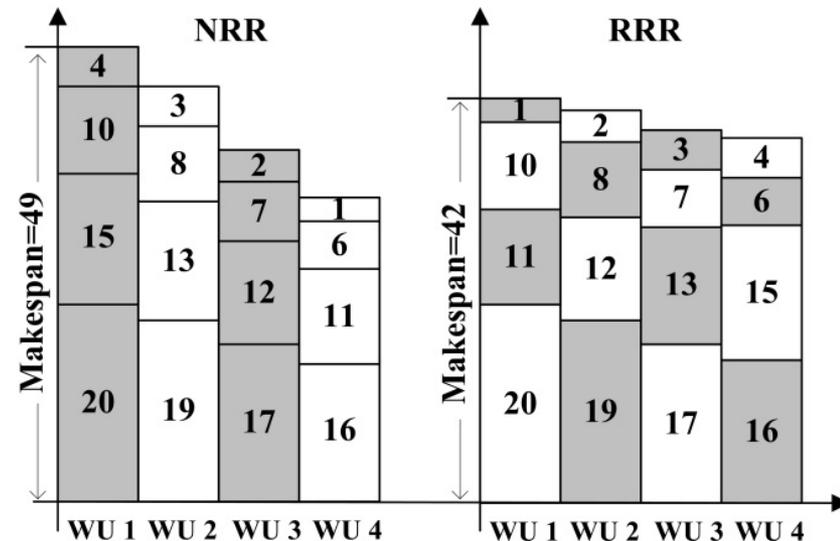
# Использование оценок вычислительной сложности

Упорядочить подзадачи в порядке убывания (оценок) сложности



1. **Прямое циклическое распределение:** подзадачи назначаются в порядке следования всем процессорам, начиная с первого, потом опять «по кругу» опять начиная с первого

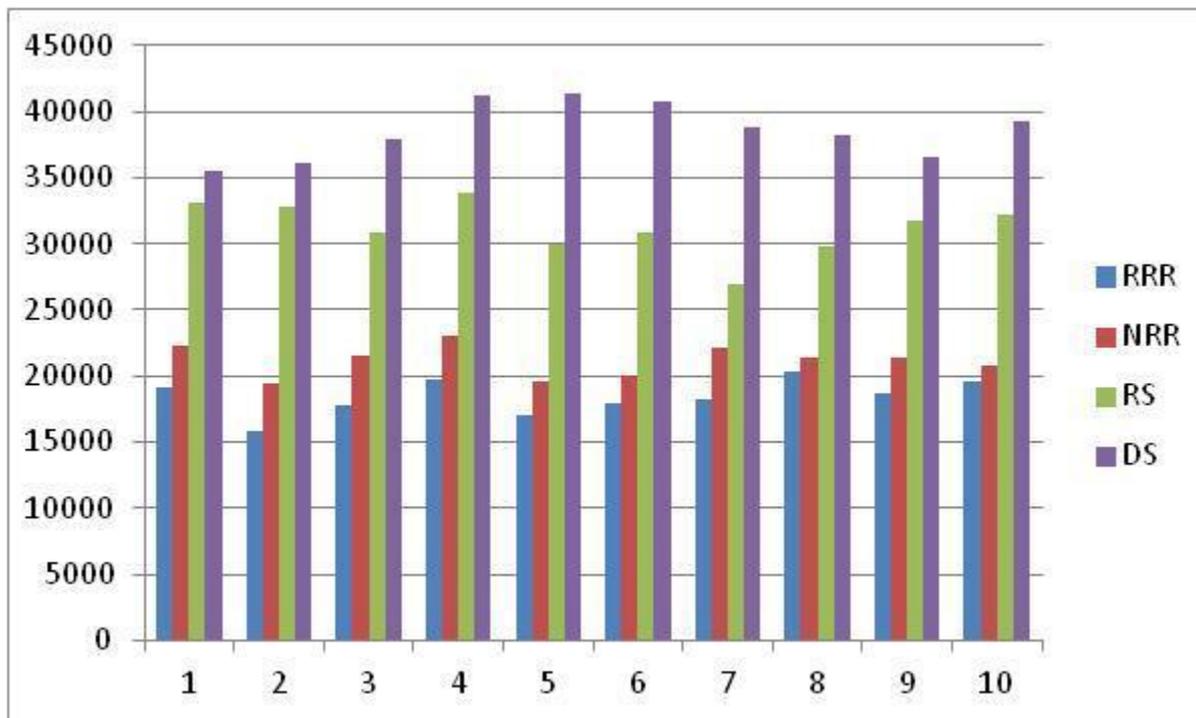
2. **Обратное циклическое распределение:** подзадачи назначаются в порядке следования всем процессорам, начиная с первого, потом начиная с последнего, потом опять с первого и т.д.



# Методы оценки сложности

- Теоретические на основе исходных данных подзадачи
  - Р.М. Колпаков, М.А. Посыпкин, И.Х. Сигал. О нижней оценке вычислительной сложности одной параллельной реализации метода ветвей и границ// Автоматика и телемеханика. 2010. № 10. С. 156-166.
- Эвристические
  - Knuth, D. E. (1975). Estimating the efficiency of backtrack programs. Mathematics of computation, 29(129), 122-136.

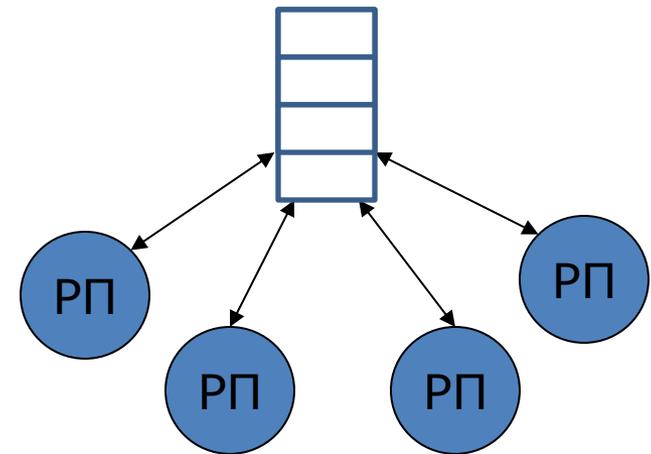
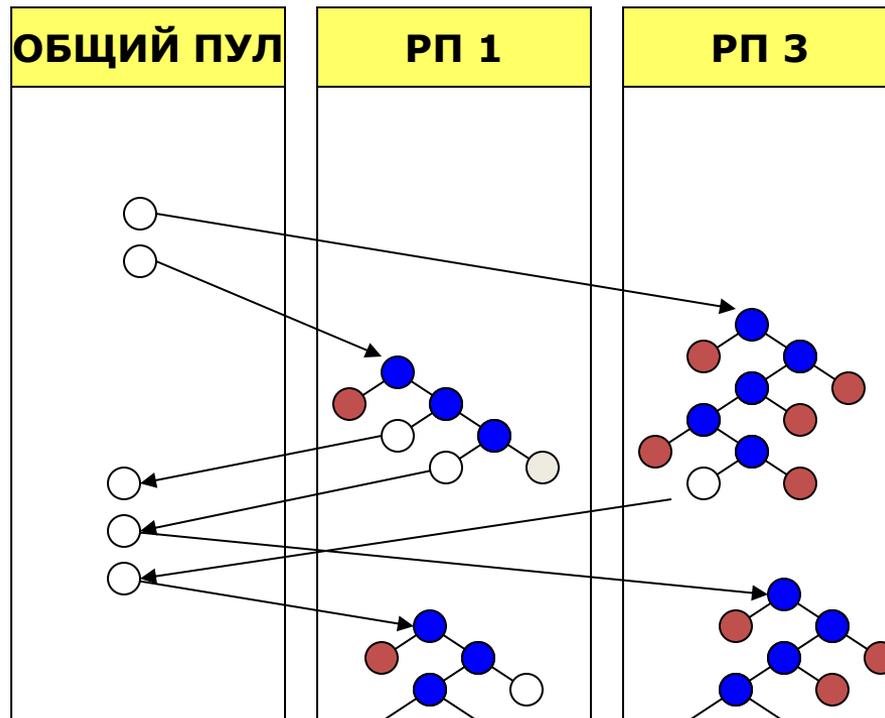
# Результаты балансировки (задача о сумме подмножеств)



*Бо Тянь, М.А. Посыпкин, И.Х. Сигал "Балансировка нагрузки на основе оценок алгоритмической сложности подзадач" // Информационные технологии и вычислительные системы, № 1, с. 10–18, 2015.*

# Реализация для общей памяти

Используются  
**POSIX Threads**



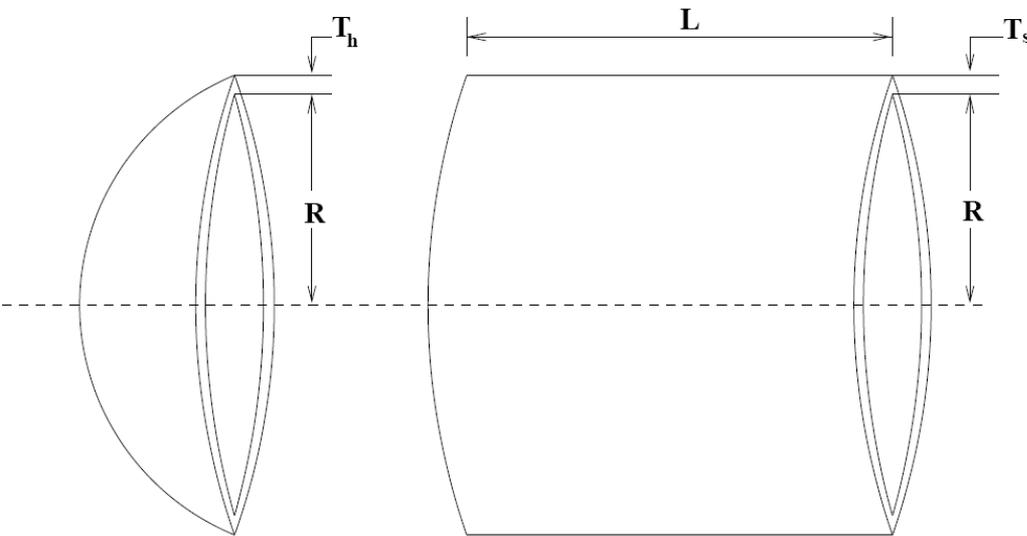
# Проблемы реализации с общей памятью

- Балансировка нагрузки
- Необходимость работы с динамической памятью: системные функции (malloc и т.п.) требуют эксклюзивности доступа.



**BNB-Solver** использует свою программно-реализованную динамическую память

# Пример: оптимальный дизайн отопительного котла



Требуется  
минимизировать затраты  
 $f(x)$  на производство при  
соблюдении  
технологических  
ограничений  $g_1$ - $g_4$ .

$$f(x) = 0.6224T_sT_hR + 1.7781T_hR^2 + 3.1661T_s^2L + 19.84T_s^2R,$$

$$g_1(x) = -T_s + 0.0193R \leq 0,$$

$$g_2(x) = -T_h + 0.00954R \leq 0,$$

$$g_3(x) = -\pi R^3L - \frac{4}{3}\pi R^3 + 1296000 \leq 0,$$

$$g_4(x) = L - 240 \leq 0.$$

# Вычислительный эксперимент для системы с общей памятью

Компьютер: 2 x 4-core Intel Xeon 5355

Число потоков	1	2	4	8
T	59.0	33.7	17.7	9.85
S	1	1.8	3.3	6.0

# Особенности распределенной среды

- Динамически меняющийся состав
- Неоднородность
- Географическая удаленность
- Ограничения на направление взаимодействия узлов

# Система FATCOP

- Предназначена для решения частично-целочисленных задач методами ветвей и отсечений
- Основана на Condor + PVM
- Позволяет решать задачи на сетях из персональных компьютеров
- *Chen Q., Ferris M. C. FATCOP: A fault tolerant Condor-PVM mixed integer programming solver //SIAM Journal on Optimization. – 2001. – Т. 11. – №. 4. – С. 1019-1036.*
- *Chen Q., Ferris M. C., Linderoth J. Fatcop 2.0: Advanced features in an opportunistic mixed integer programming solver //Annals of Operations Research. – 2001. – Т. 103. – №. 1-4. – С. 17-32.*

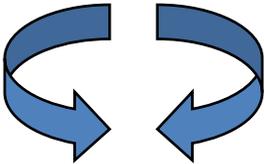
# Condor and PVM



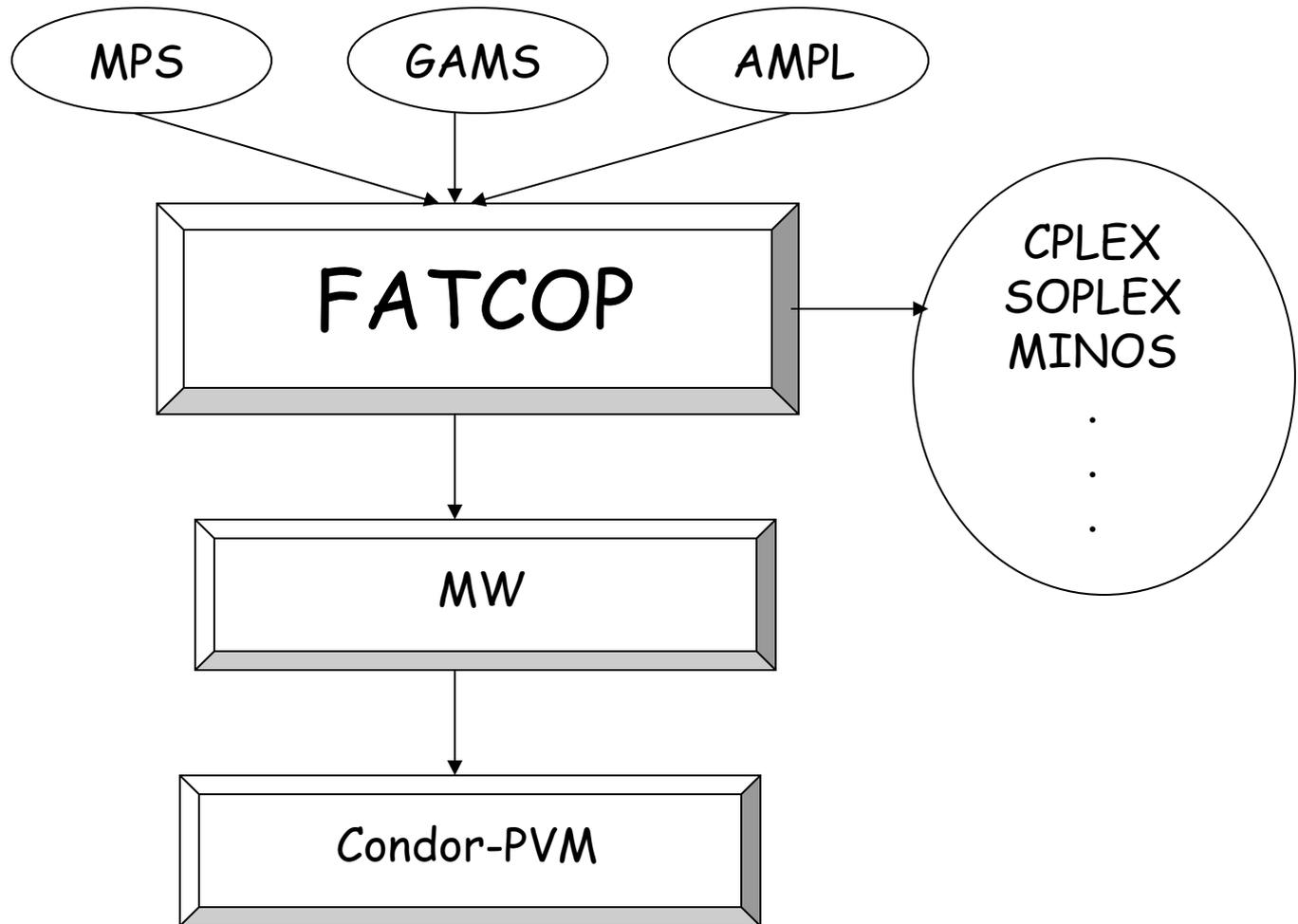
- **Condor**: a distributed resource management system
- Manages large **heterogeneous** clusters of UNIX workstations
- Design motivated to use 'idle' capacity for long-running, computation-intensive jobs



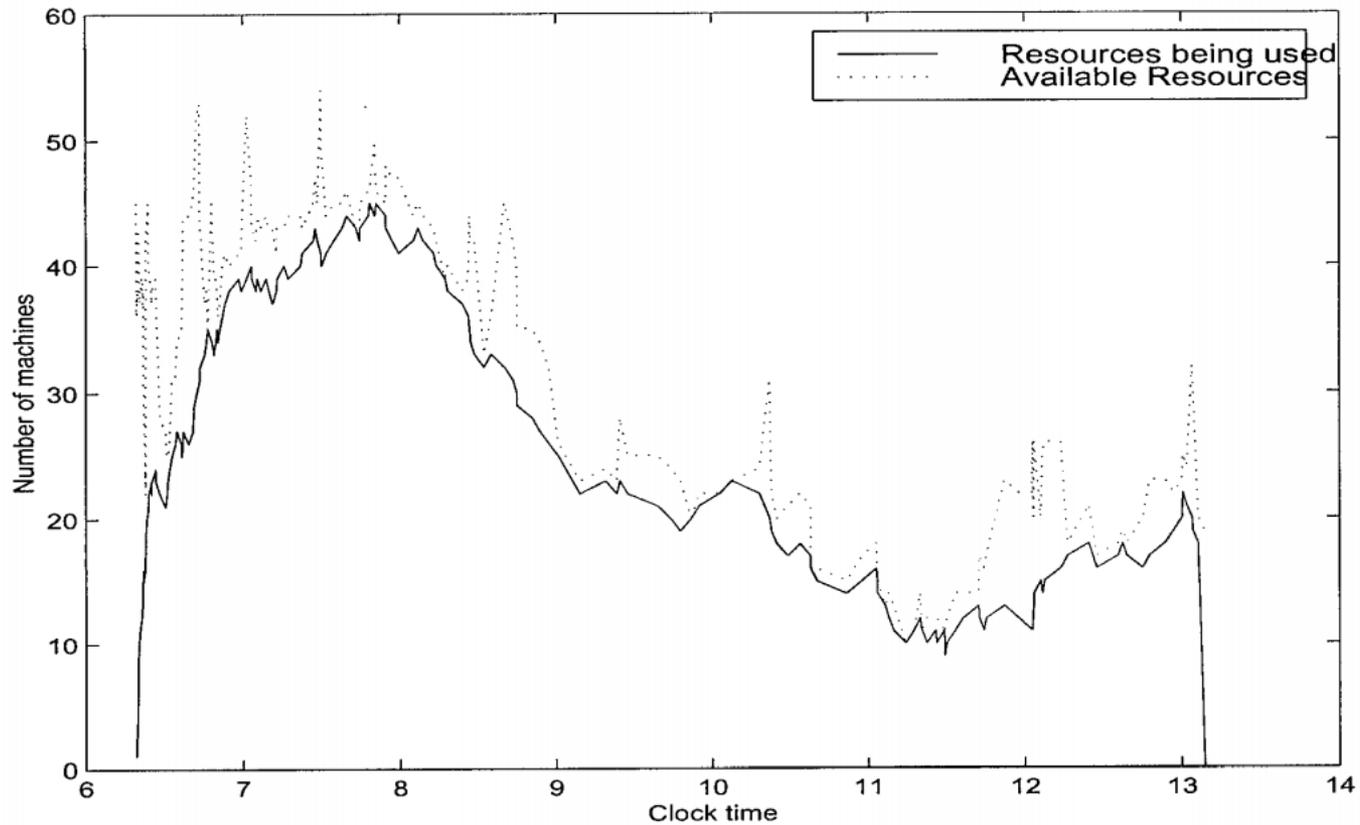
- **PVM (parallel virtual machine)**: software allowing heterogeneous network of computers to appear as a single concurrent computational resource
- Unified framework for parallel program development



- **PVM and Condor** perfectly matched
- Framework to run parallel applications in a distributed opportunistic environment



# Использование ресурсов



# Приложение vs библиотека

- **ПОДХОД ПРИЛОЖЕНИЙ**: для каждой новой проблемы писать новую программу
- **БИБЛИОТЕКА**: выделить и реализовать общие части, которые затем использовать повторно для каждой новой задачи

# ПРИЛОЖЕНИЕ vs БИБЛИОТЕКА ДЛЯ РАСПРЕДЕЛЕННЫХ ПРИЛОЖЕНИЙ

«ПОСЛЕДОВАТЕЛЬНЫЕ»  
МАШИНЫ



ПАРАЛЛЕЛЬНЫЕ И  
РАСПРЕДЕЛЕННЫЕ  
КОМПЛЕКСЫ

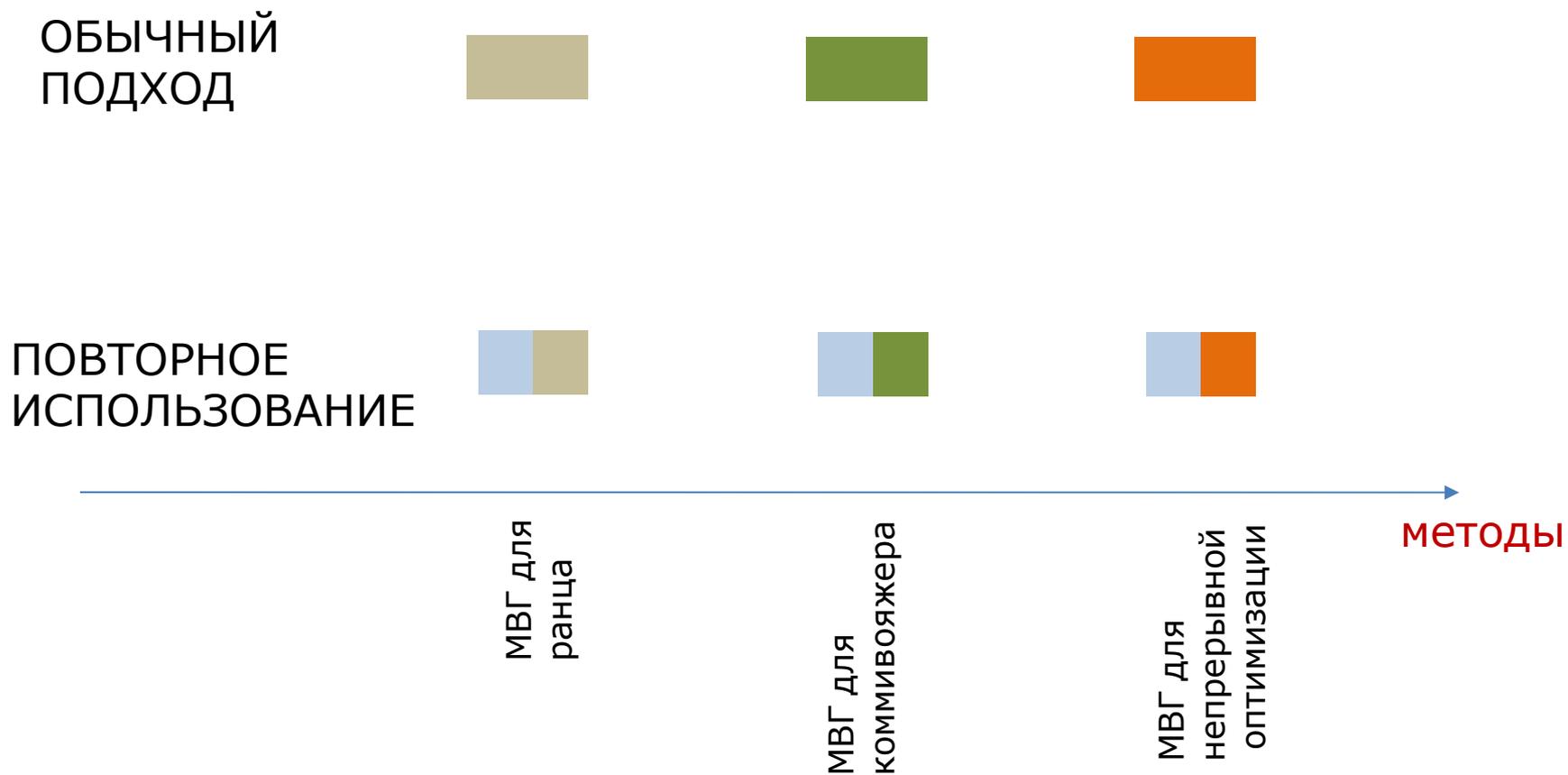


Общая часть больше, т.к.  
платформа сложнее

# Дополнительная литература

- Strongin R. G., Sergeyev Y. D. Global optimization with non-convex constraints: Sequential and parallel algorithms. – Springer Science & Business Media, 2013. – Т. 45.
- Стронгин, Р. Г., Гергель, В. П., Гришагин, В. А., Баркалов, К. А. (2013). Параллельные вычисления в задачах глобальной оптимизации—М. Издательство Московского университета.—2013.
- Gergel, V. P., & Strongin, R. G. (2005). Parallel computing for globally optimal decision making on cluster systems. *Future Generation Computer Systems*, 21(5), 673-678.
- Barkalov, K., Gergel, V., & Lebedev, I. (2015). Use of Xeon Phi Coprocessor for Solving Global Optimization Problems. In *Parallel Computing Technologies* (pp. 307-318). Springer International Publishing.

# ПОВТОРНОЕ ИСПОЛЬЗОВАНИЕ ПРОБЛЕМНО-НЕЗАВИСИМОЙ ЧАСТИ



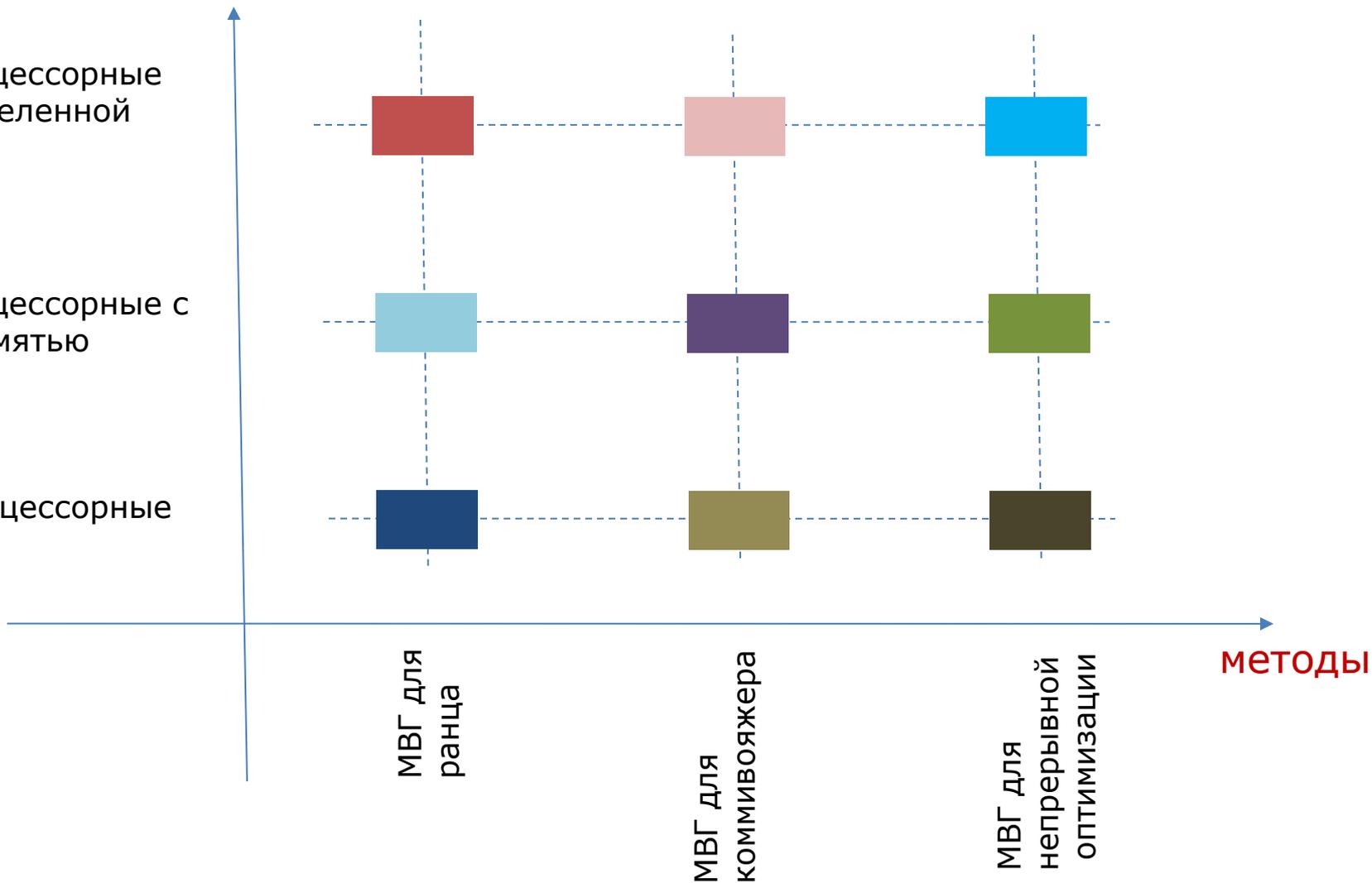
# РЕАЛИЗАЦИЯ ДЛЯ РАЗЛИЧНЫХ АРХИТЕКТУР: ОБЫЧНЫЙ ПОХОД

архитектуры

многопроцессорные  
с распределенной  
памятью

многопроцессорные с  
общей памятью

однопроцессорные



# РЕАЛИЗАЦИЯ ДЛЯ РАЗЛИЧНЫХ АРХИТЕКТУР: ПОВТОРНОЕ ИСПОЛЬЗОВАНИЕ

