Методы обеспечения отказоустойчивости, основанные на координированном сохранении контрольных точек

А.А. Бондаренко, П.А. Ляхов, М.В. Якобовский

ИПМ им. М.В.Келдыша РАН

Суперкомпьютерные дни в России
25-26 сентября 2017 г. Москва

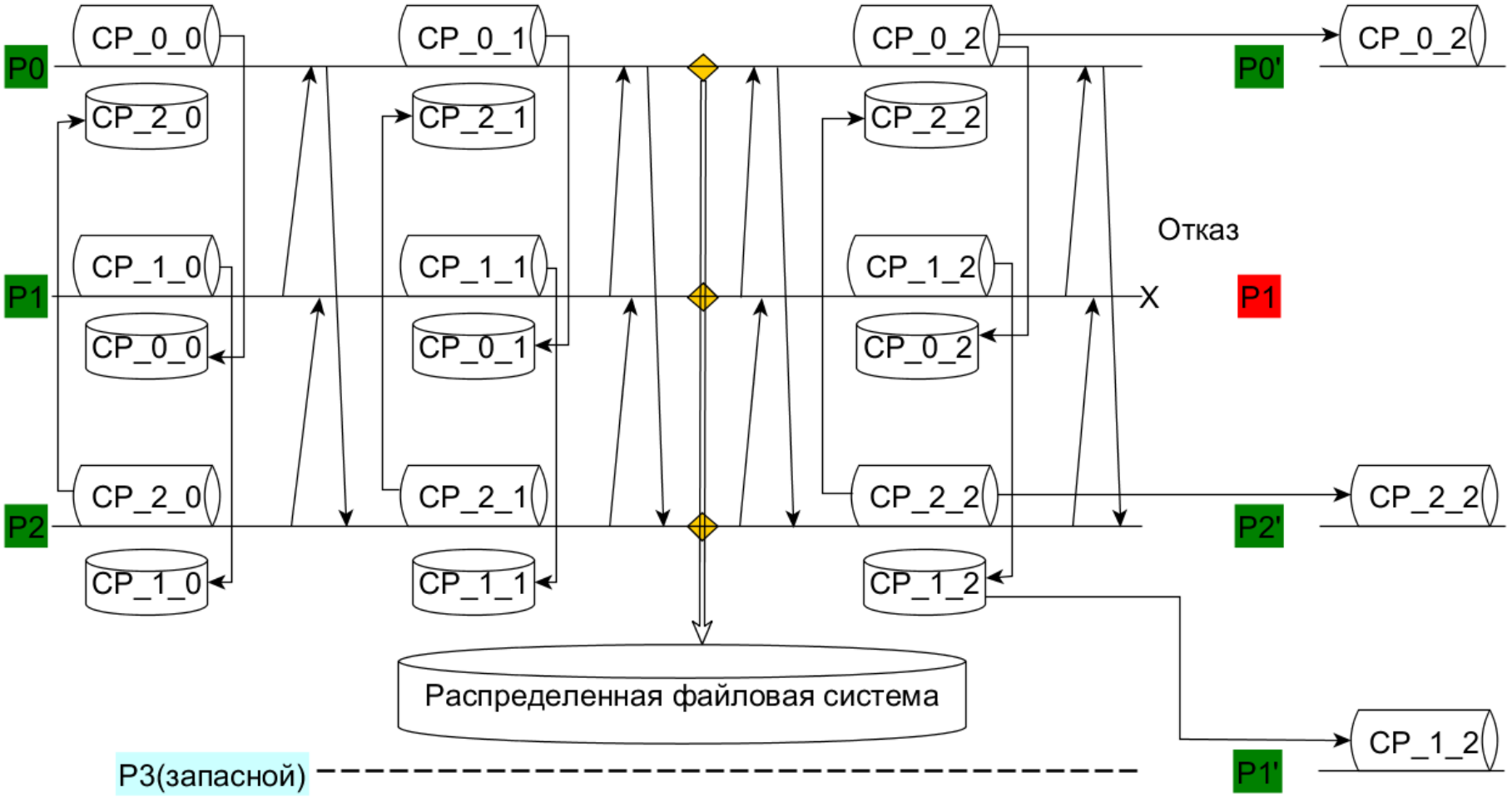
Важные работы

Год	Источники	Особенности
1993	Charm++ http://charmplusplus.org/	язык программирования и среда исполнения параллельных программ
1996/2002	E.N. Elnozahy, et al. A Survey of Rollback-Recovery Protocols in Message-Passing Systems	разбираются вопросы организации координированного и некоординированного сохранения контрольных точек
2003	Cornell Checkpoint pre-Compiler	компилятор, позволяющий сохранять контрольные точки на уровне пользователя
2005	controller/compiler for Portable Checkpointing http://cppc.des.udc.es	инструмент сохранения контрольных точек для гетерогенных кластеров и грид-систем, с использованием переносимых: протоколов, контрольных точек и кода.
2005	BLCR http://crd.lbl.gov/departments/computer-science/CLaSS/research/BLCR/	библиотека реализованная на уровне системы, позволяет сохранять процессы в файл, а в случае отказа восстанавливать процессы из этого файла

Важные работы

Год	Источники	Особенности
2008	P.M. Kogge, et al. ExaScale Computing Study: Technology Challenges in Achieving Exascale Systems — Tech. Report TR-2008-13.	отчет посвященный списку задач, требующих решения для построения и использования вычислительных машин уровня Экзаскейл
2012	ULFM http://fault-tolerance.org/ Продолжение FT-MPI (с 2000 г.) http://icl.cs.utk.edu/ftmpi/index.html	набор дополнительных для MPI функций, которые позволят восстанавливать MPI среду после отказа
2014	Di, Sheng, et al. "Optimization of multi-level checkpoint model for large scale HPC applications." IEEE, 2014.	одна из важных статей в которой описывается метод многоуровневого сохранения контрольных точек
2014	M. Besta, T. Hoefler "Fault Tolerance for Remote Memory Access Programming Models" // (HPDC'14), ACM, Jun. 2014.	особый способ обеспечения отказоустойчивости основанный на удаленном доступе к памяти процессов (RMA)

Многоуровневая стратегия, основанная на контрольных точках



Теоретическая модель

Полное время работы программы можно оценить по формуле

$$T_{final} = T_e + C_1(x_1 - 1) + C_2(x_2 - 1) + \left(\frac{T_e}{2x_1} + D + R_1\right)n_1 + \left(\frac{T_e + C_1 x_1}{2x_2} + D + R_2\right)n_2$$

- T_e - время работы программы без обеспечения отказоустойчивости.

Для *i*-го уровня отказов:

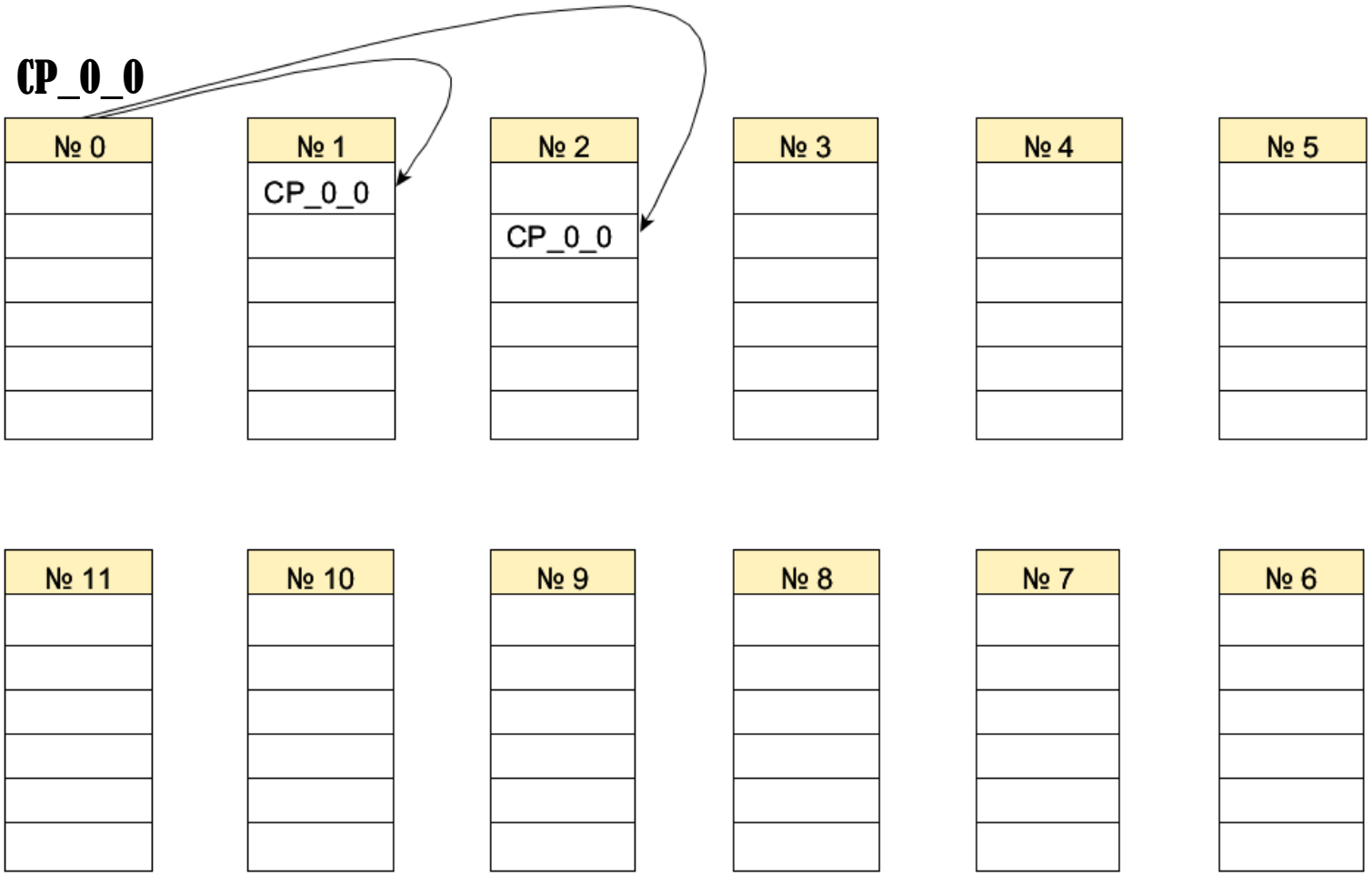
- за время расчёта происходит (x_i-1) сохранений контрольных точек,
- C_i секунд занимает процесс записи контрольной точки.

За время расчёта происходит n_i отказов, каждый из которых влечёт за собой:

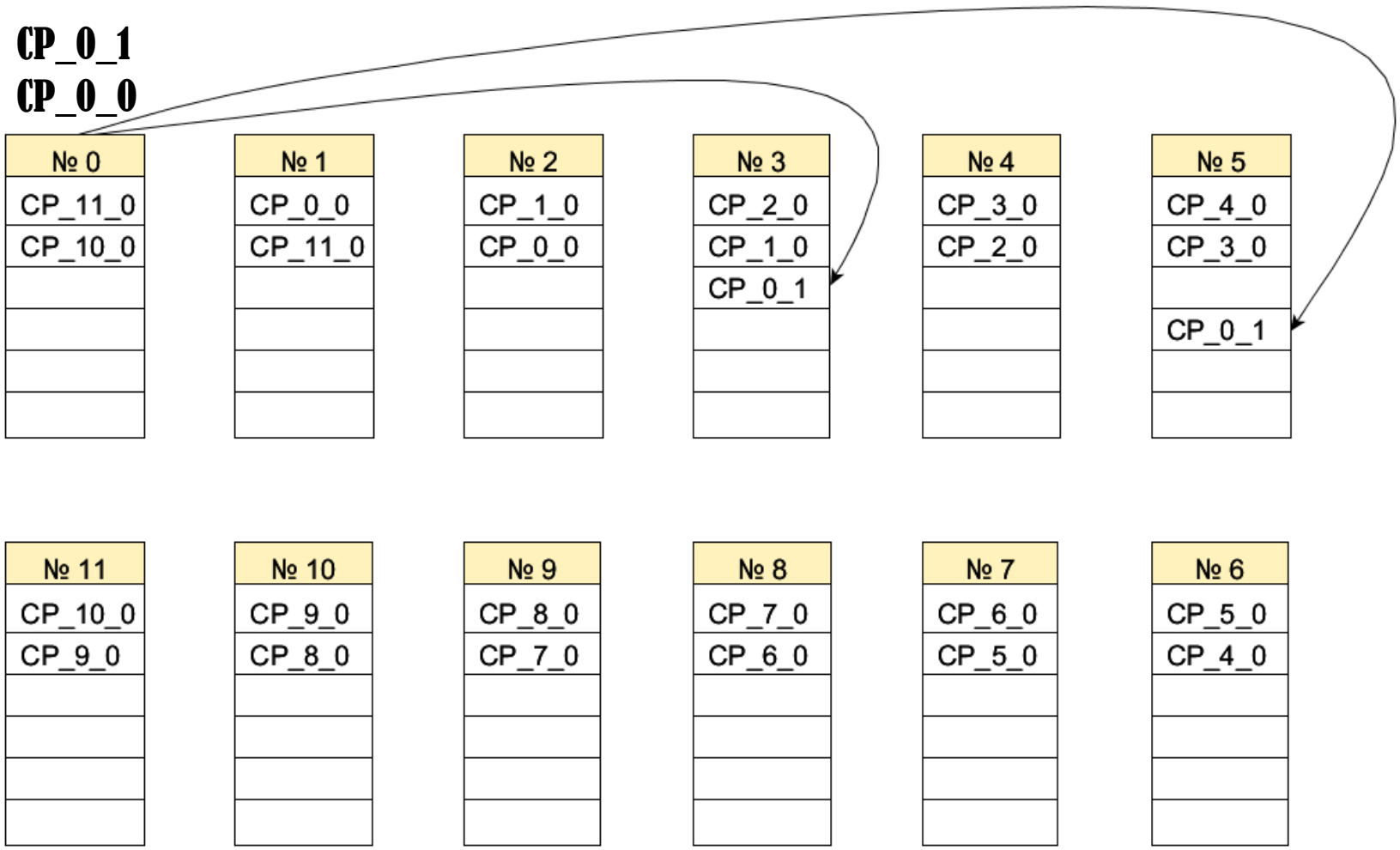
- восстановление системы *D* (определение функционирующих элементов системы, восстановление параллельной среды выполнения и др.),
- затраты на чтение контрольной точки *i*-го уровня R_i,
- затраты на перерасчёт потерянных в результате отказа вычислений.

Di S., Bouguerra M.S.; Bautista-Gomez L., Cappello F.; ,Optimization of multi-level checkpoint model for large scale HPC applications // Parallel and Distributed Processing Symposium, 2014 IEEE 28th International P. 1181-1190, 2014.

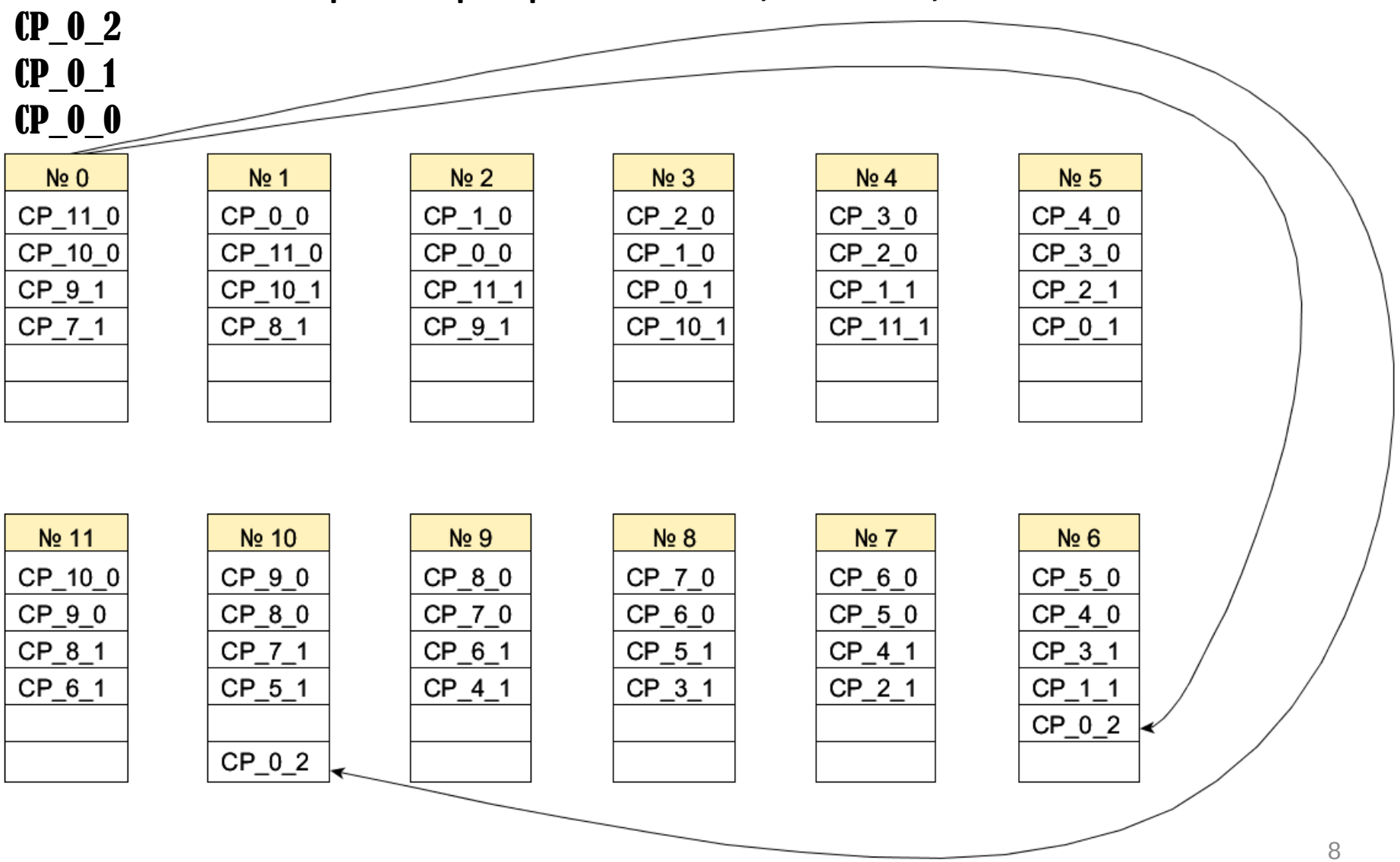
Метод локального дублирования
Пример при N = 12, DF = 2, SD = 3



Метод локального дублирования
Пример при N = 12, DF = 2, SD = 3



Метод локального дублирования
Пример при N = 12, DF = 2, SD = 3



Метод локального дублирования
Пример при N = 12, DF = 2, SD = 3

№ 0
CP_11_0
CP_10_0
CP_9_1
CP_7_1
CP_6_2
CP_2_2

№ 1
CP_0_0
CP_11_0
CP_10_1
CP_8_1
CP_7_2
CP_3_2

№ 2
CP_1_0
CP_0_0
CP_11_1
CP_9_1
CP_8_2
CP_4_2

№ 3
CP_2_0
CP_1_0
CP_0_1
CP_10_1
CP_9_2
CP_5_2

№ 4
CP_3_0
CP_2_0
CP_1_1
CP_11_1
CP_10_2
CP_6_2

№ 5
CP_4_0
CP_3_0
CP_2_1
CP_0_1
CP_11_2
CP_7_2

№ 11
CP_10_0
CP_9_0
CP_8_1
CP_6_1
CP_5_2
CP_1_2

№ 10
CP_9_0
CP_8_0
CP_7_1
CP_5_1
CP_4_2
CP_0_2

№ 9
CP_8_0
CP_7_0
CP_6_1
CP_4_1
CP_3_2
CP_11_2

№ 8
CP_7_0
CP_6_0
CP_5_1
CP_3_1
CP_2_2
CP_10_2

№ 7
CP_6_0
CP_5_0
CP_4_1
CP_2_1
CP_1_2
CP_9_2

№ 6
CP_5_0
CP_4_0
CP_3_1
CP_1_1
CP_0_2
CP_8_2

Метод локального дублирования

Введем параметр $j \in \{1, 2, ..., DF\}$ для обозначения номера передаваемой копии локальной контрольной точки с *i–ого* узла. Тогда на *k–ой* итерации сохранения, запись *j–ой* копии с *i–ого* узла должна быть осуществлена в память вычислительного узла с номером, определяемым формулой:

$$receiver(i, j, k) = [i + j*DF^{(k)mod(SD)} + (k)mod(SD)]mod(N) \quad (1)$$

Утверждение

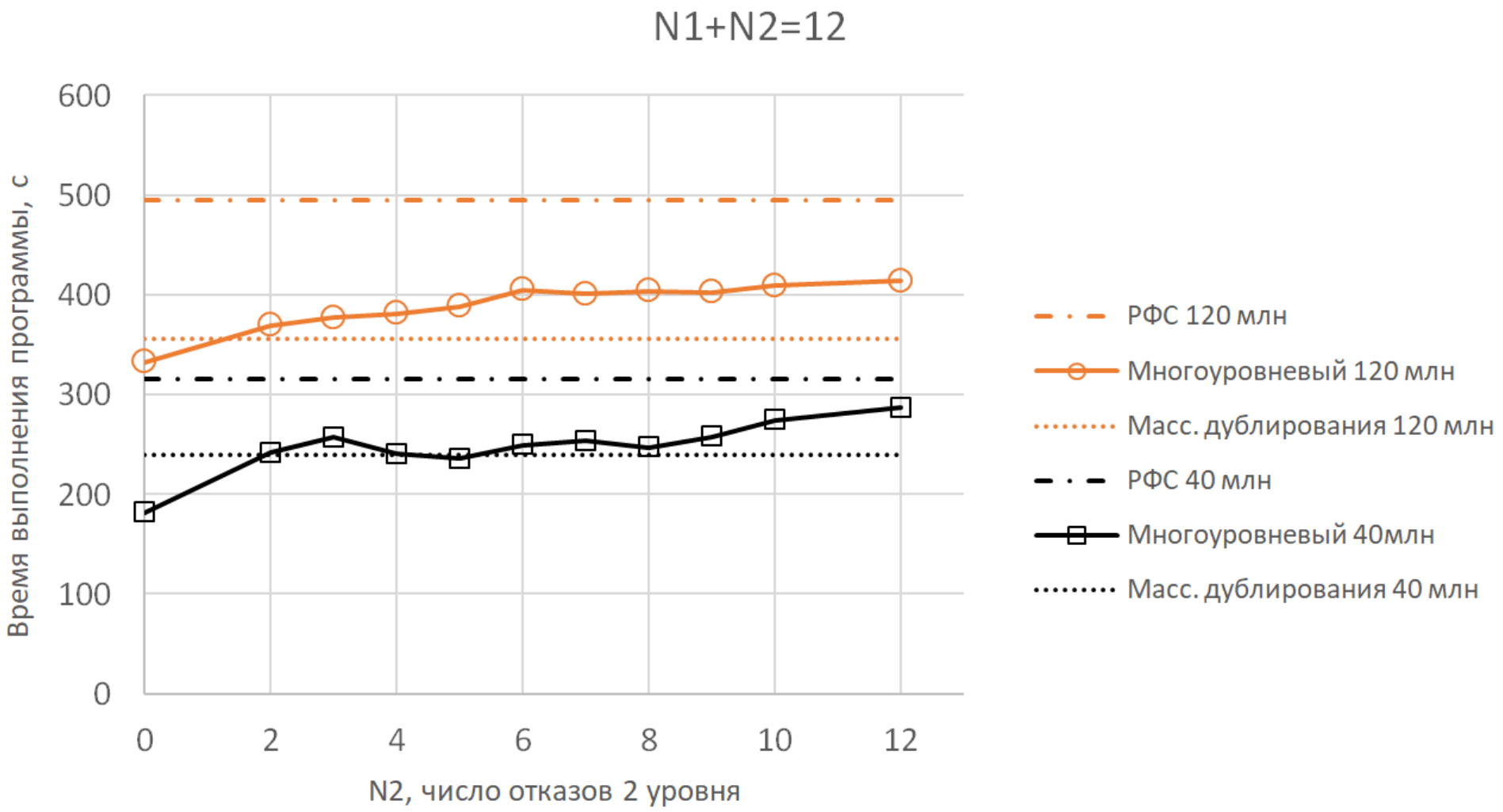
В данной схеме объем сохраняемой на каждом узле информации можно оценить величиной $V = V_0 * SD * (DF+1)$, где V_0 – средний объем локальных контрольных точек. Пусть в вычислительной системе достаточно много узлов $N \geq DF^{SD} + SD$ и число отказавших узлов не превосходит

$$(DF - 1) * SD + 1 \quad (2)$$

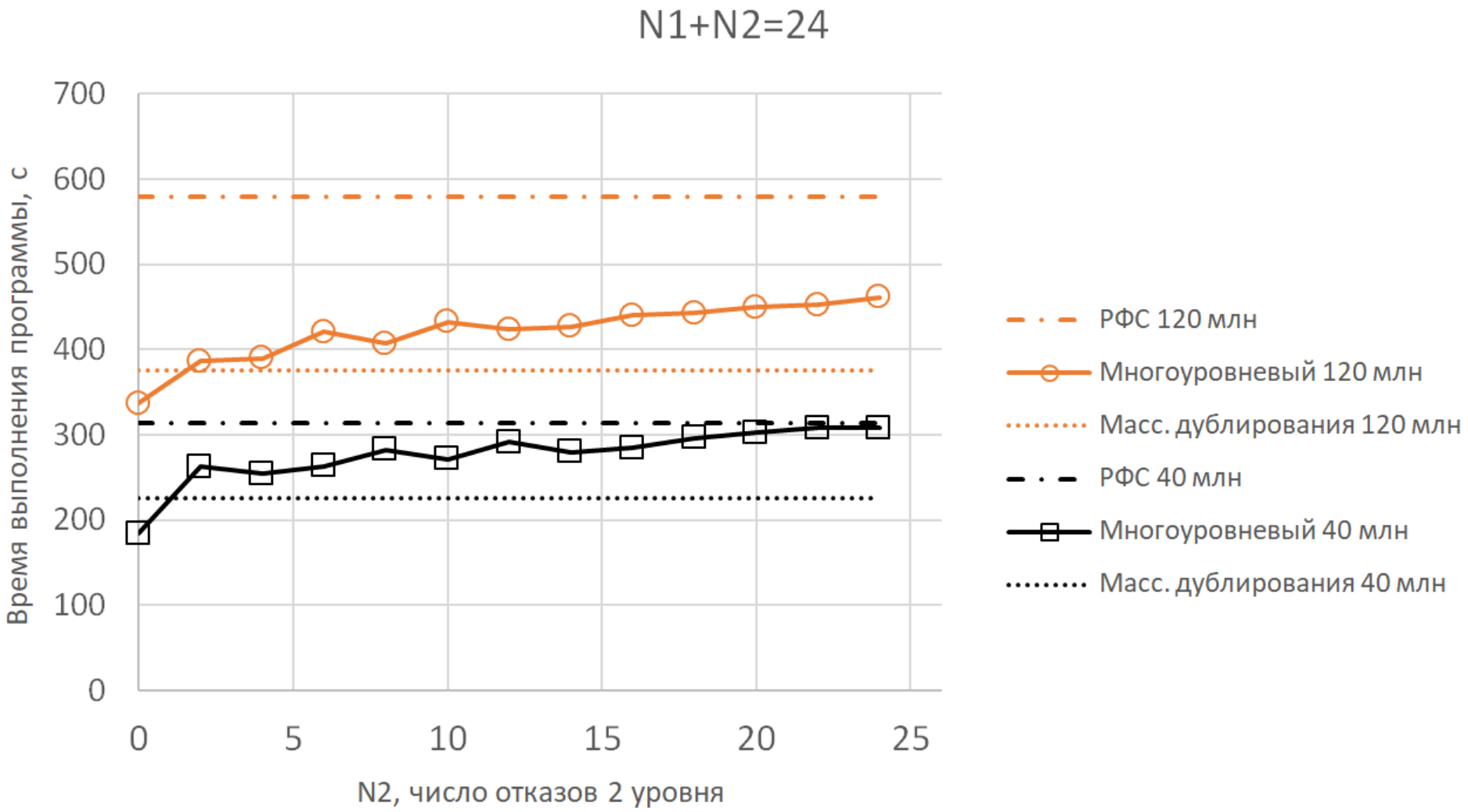
Тогда, после первых SD итерации сохранения, в системе будут доступны локальные контрольные точки MPI-процессов со всех узлов одной из последних SD итераций сохранения, то есть согласованная глобальная контрольная точка.

Бондаренко А. А., Якобовский М. В. Обеспечение отказоустойчивости высокопроизводительных вычислений с помощью локальных контрольных точек //Вестник Южно-Уральского государственного университета. Серия: Вычислительная математика и информатика. – 2014. – Т. 3. – №. 3.

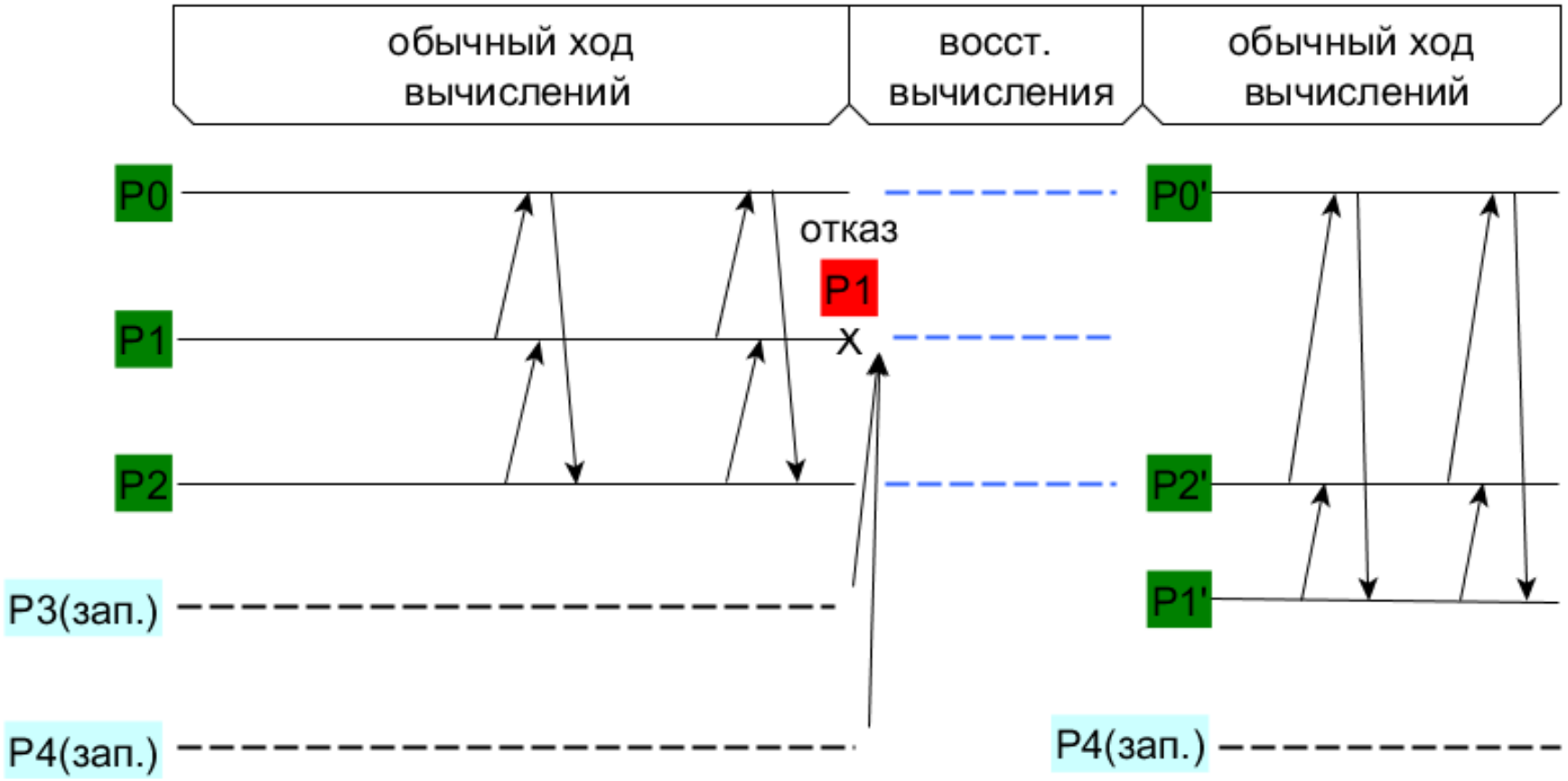
Сравнение накладных расходов



Сравнение накладных расходов

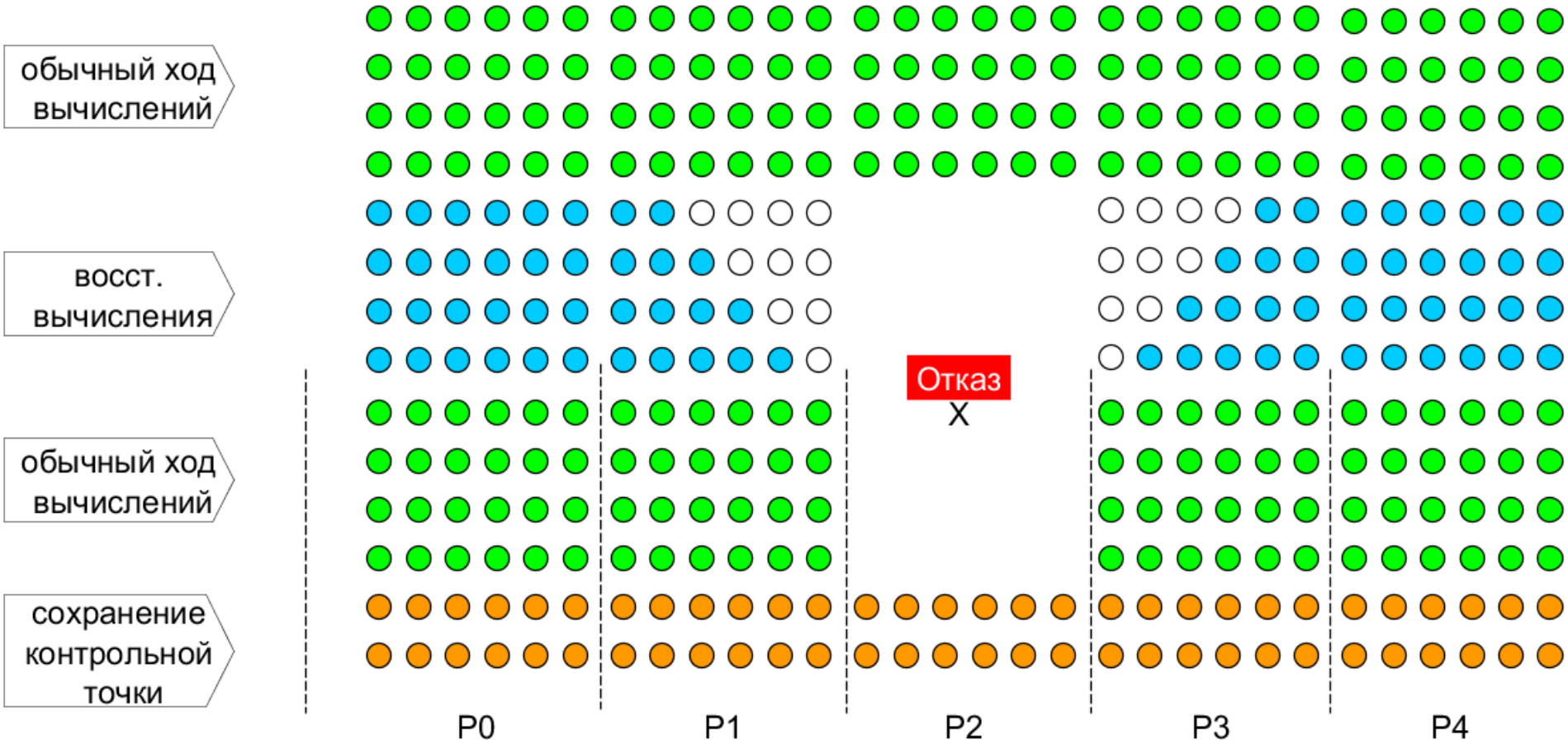


Метод пересчета

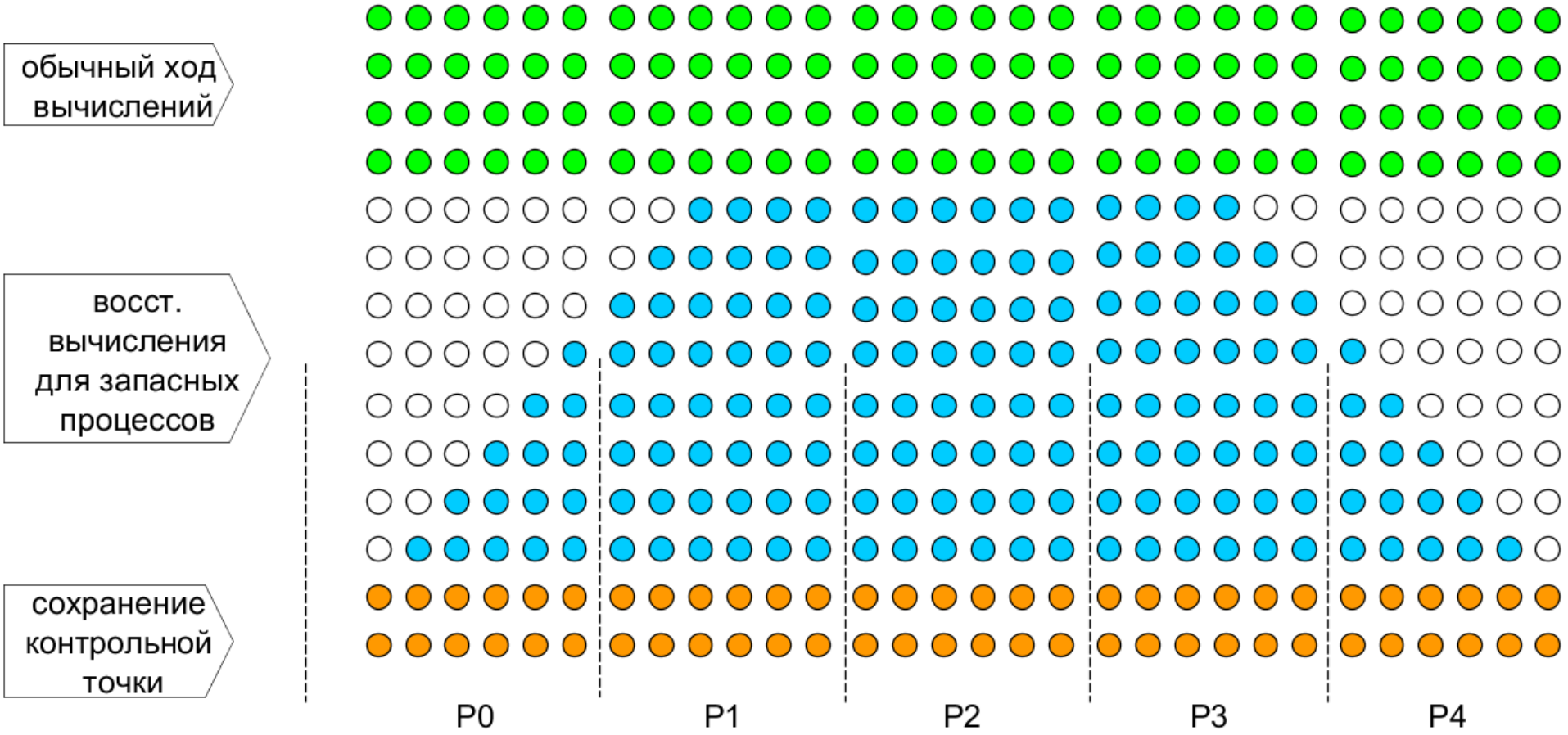


Четверушкин Б. Н., Якобовский М. В. Вычислительные алгоритмы и отказоустойчивость гиперэкзафлопсных вычислительных систем //Доклады Академии наук. – Федеральное государственное унитарное предприятие Академический научно-издательский, производственно-полиграфический и книгораспространительский центр Наука, 2017. – Т. 472. – №. 1. – С. 13-17.

Метод пересчета



Метод пересчета



Метод пересчета

Рассмотрим одномерное гиперболическое уравнение

$$\frac{\partial^2 \Phi}{\partial x^2} - \frac{1}{C^2}\frac{\partial^2 \Phi}{\partial t^2} = F(x,t)$$

γ – число куранта, тогда примем $\alpha = 1 + 2\gamma\frac{k_1}{n_0}$

k_1 – число итераций с момента последнего сохранения контрольной точки

n_0 – число узлов сетки одного процесса в начальной постановке задачи

Тогда p – число дополнительных процессов используемых для пересчета

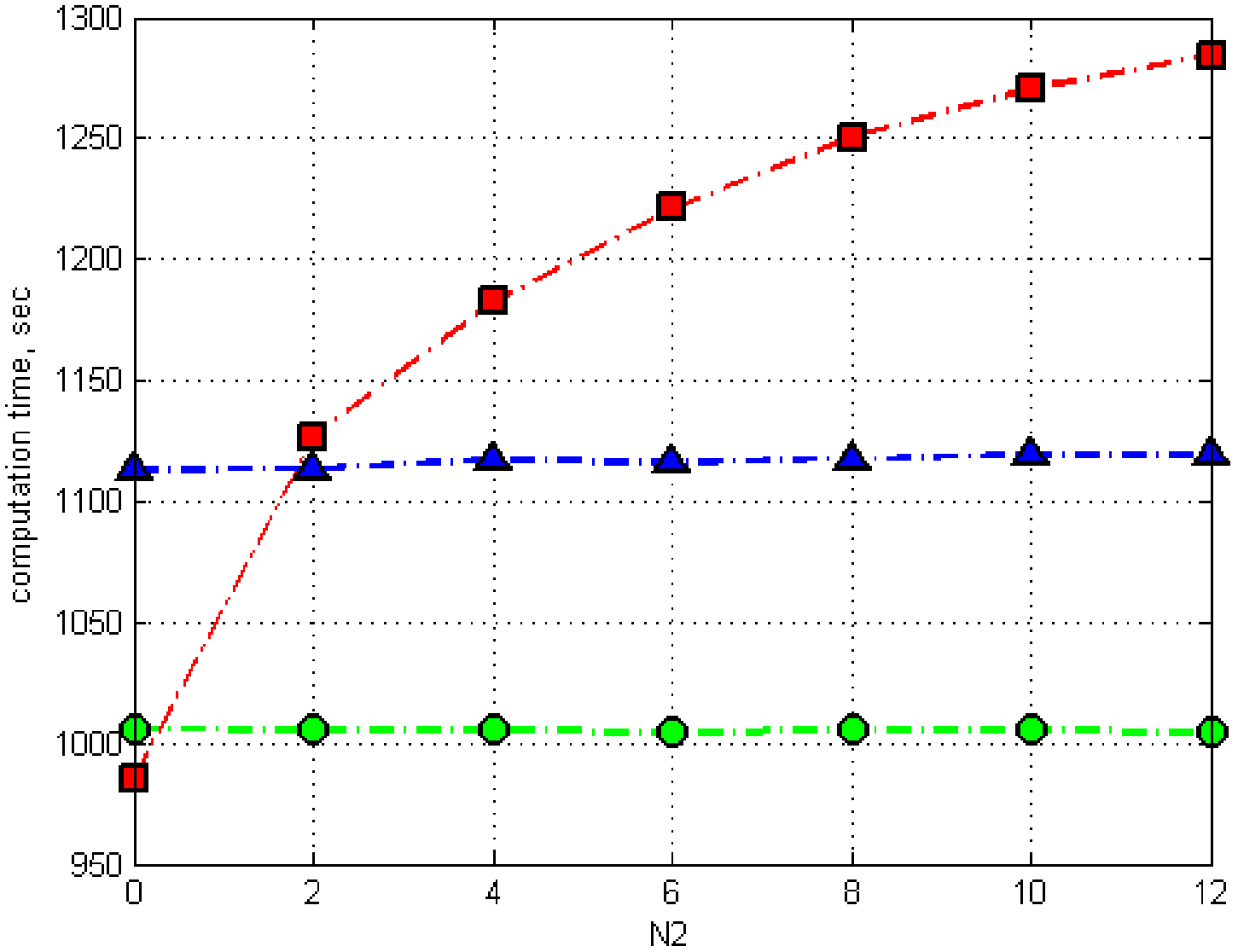
$$p \geq 2\alpha$$

Для двумерных и трехмерных вычислений выполняется следующая оценка

$$p_{2D} \geq \frac{2}{3}(1 + \alpha + \alpha^2)$$

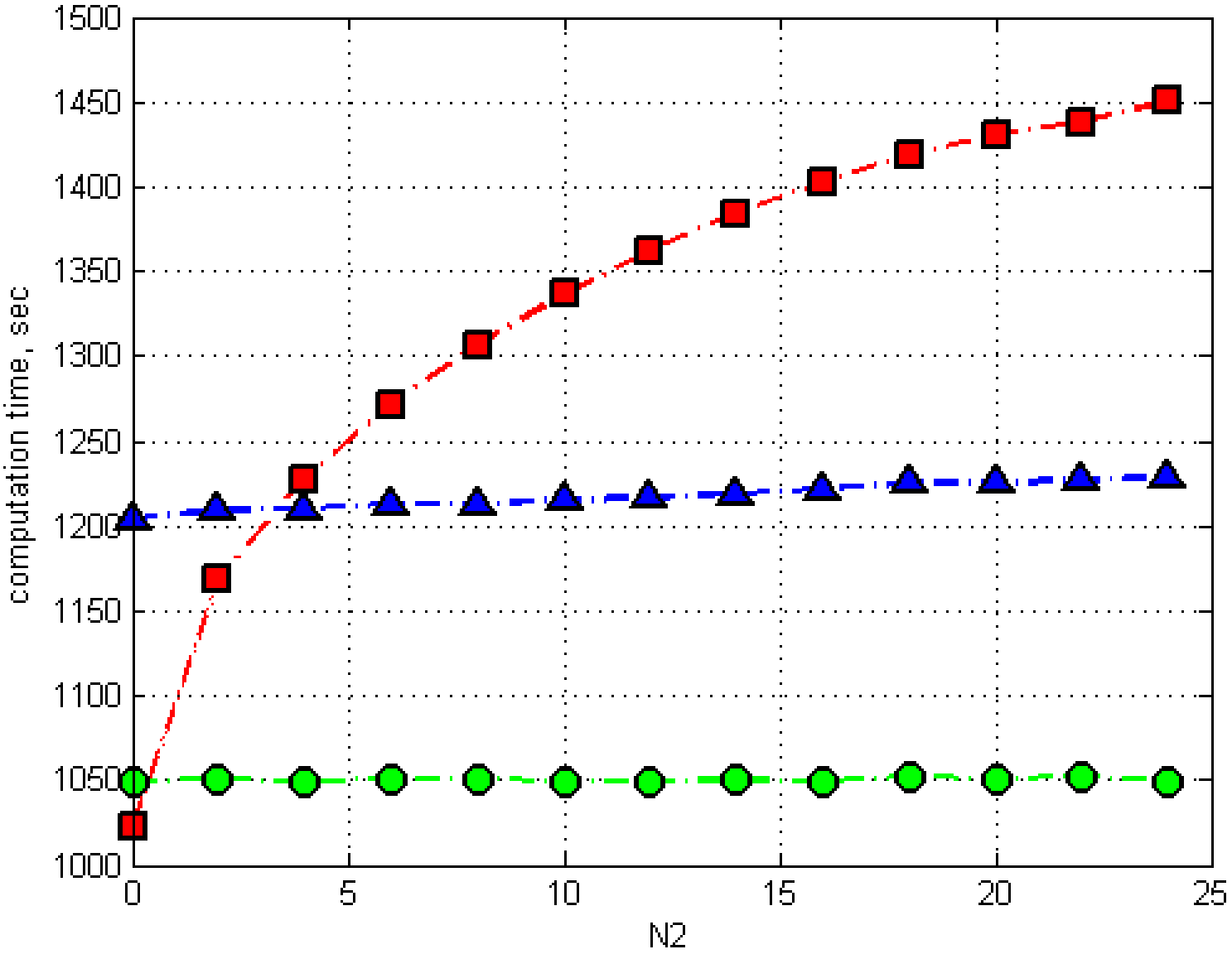
$$p_{3D} \geq \frac{1}{2}(1 + \alpha + \alpha^2 + \alpha^3)$$

N1+N2 = 12



N1 - число единичных отказов, N2 – число кратных отказов

N1+N2 = 24



N1 - число единичных отказов, N2 – число кратных отказов

Заключение

Для будущих Суперкомпьютеров эффективность организации отказоустойчивых вычислений существенно зависит от:

- архитектуры Суперкомпьютеров (топологии связи, размера и скорости работы локальных устройств хранения информации, и пр.),
- особенностей используемых параллельных алгоритмов (локальности данных, объема данных необходимых для восстановления расчета, сложности реализации этапа восстановительного пересчета и пр.),
- наличия дополнительной информации о частоте и кратности отказов.

Спасибо

Вопросы?