ПОВЫШЕНИЕ НАДЕЖНОСТИ ГЕТЕРОГЕННЫХ РАСПРЕДЕЛЕННЫХ ОТКАЗОУСТОЙЧИВЫХ СИСТЕМ ЗА СЧЕТ ГРУППИРОВКИ ВЫЧИСЛИТЕЛЬНЫХ УЗЛОВ

А.Н. Фирсов

Аннотация

В статье предлагаются формула и алгоритм для оценки надежности устойчивых к произвольным отказам систем, работающих в гетерогенной по надежности среде. Демонстрируются ситуации, когда существующие решения неэффективны и предлагается способ, позволяющий повысить надежность существующих решений за счет группировки вычислительных узлов.

1. Введение

С течением времени зависимость людей от техники только увеличивается, а сама техника усложняется. Поэтому остро встает проблема обеспечения отказоустойчивости техники. Так если вероятность сбоя отдельно взятого компьютера невелика, то в распределенных системах, состоящих из тысяч или даже миллионов таких компьютеров, сбои происходят постоянно.

Византийская модель сбоев [1] (или модель произвольных сбоев) является наиболее общей, включающей в себя все остальные. Она не накладывает никаких ограничений на действия сбойных процессов и, благодаря этому, позволяет моделировать такие типы сбоев, как поломка оборудования, потеря связи, неисправность оборудования, программные сбои, ошибки операторов и действия злоумышленников.

Особый интерес представляют асинхронные распределенные системы, т.к. они не накладывают никаких временных ограничений, и, благодаря этому, идеально подходят для работы в сети интернет. Наиболее универсальной моделью вычислительного узла является машина с конечным числом состояний (state machine).

В данной работе рассматриваются асинхронные распределенные системы, которые, во-первых, устойчивы к произвольным отказам, во-вторых, позволяют выполнять прикладные программы, требующие сохранять свое состояние между обращениями к ним, в-третьих, обеспечивают отказоустойчивость на программном уровне без необходимости в какой-либо специальной аппаратуре и, в-четвертых, делают все это прозрачно для прикладных программ.

2. Проблемы существующих решений

В настоящее время существует несколько систем, отвечающих приведенным выше требованиям: Rampart[2], SecureRing[3], BFT[4] и т.д.. Все они реализуют единственный известный на сегодня программный метод защиты от произвольных сбоев – активную репликацию. Суть этого метода заключается в том, что параллельно выполняются нескольких копий (или, как их еще называют, реплик) одной и той же программы.

Во всех этих системах и практически во всех других работах, связанных с устойчивостью к произвольным отказам [1-8], фиксируется некоторое число f – максимальное количество процессоров (или вычислительных узлов), которое может выйти из строя из имеющихся n процессоров, не приведя к сбою системы в целом, и уже на его основе доказываются все утверждения. Это удобно для проектирования и доказательства корректности алгоритмов, однако в действительности не может быть никакой гарантии, что из строя не выйдет большее число процессоров, просто такая вероятность считается пренебрежимо малой и не учитывается.

Ни одна из приведенных выше систем не предоставляет алгоритма для оценки вероятности сбоя получаемого решения, а именно в этом показателе больше всего заинтересован пользователь отказоустойчивых систем. Как правило, предоставляется лишь возможность управлять необходимым уровнем надежности, изменяя параметр f. Увеличивая его для повышения надежности или уменьшая для экономии ресурсов. Такой подход в неявном виде предполагает, что все вычислительные узлы одинаковы по надежности. В сети же Интернет и распределенных системах, строящихся на ее основе, это требования, как правило, не выполняется. В одну распределенную систему могут быть объединены как сервера с устройствами бесперебойного питания, находящиеся под управлением профессиональных администраторов, с надежными и обновляемыми операционными системами и антивирусным программным обеспечением, так и рабочие станции, зараженные вирусами, с ненадежной аппаратурой и уязвимой операционной системой. Очевидно, что вероятность их отказа может отличаться на несколько порядков. А именно на использование в сети интернет и в асинхронных распределенных системах ориентируются вышеперечисленные системы обеспечения отказоустойчивости.

Пренебрежение тем фактом, что вычислительные узлы могут значительно отличаться по надежности может повлечь ситуацию, когда увеличение параметра f приведет к уменьшению надежности системы в целом, что будет продемонстрировано ниже.

3. Оценка надежности вычислительных узлов на основе собираемых статистических данных

Очевидно, что на надежность отказоустойчивой системы в целом влияет надежность ее составных частей, и если надежности этих составных частей различаются, то необходимо знать каждую из них, в отличие от случая, когда они одинаковы, – тогда достаточен один агрегированный параметр.

Наиболее удобным на практике способом получения значений надежности вычислительных узлов является сбор статистики об их отказах в течение достаточно длительного промежутка времени. Этому вопросу посвящен четвертый раздел.

Чем реже вычислительный узел выходит из строя, тем он надежнее. Рассмотрим время между отказами одного вычислительного узла как случайную величину. Т.к. износом оборудования можно пренебречь, нет других факторов, влияющих на сбои и имеющих накопительный характер, и нет доминантного фактора, то случайная величина имеет экспоненциальный закон распределения. По свойствам экспоненциального распределения, вероятность отказа вычислительного узла в течение двух любых, одинаковых по длине, промежутков времени одинакова. Поэтому вероятность отказа в любой единичный промежуток времени одинакова и ее можно использовать в качестве характеристики надежности (чем меньше вероятность отказа за единицу времени, тем надежнее вычислительный узел).

Обозначим за $\beta_i (i=\overline{1,n})$ вероятность возникновения сбоя на i-том вычислительном узле в некоторый единичный промежуток времени. Для упрощения получаемых формул этот единичный промежуток выбирается равным времени, которое необходимо отказоустойчивой системе для ликвидации последствий отказа. Его значение зависит от используемых алгоритмов обеспечения отказоустойчивости и восстановления, а так же от времени, которое требуется на передачу сообщений.

Значение β_i может быть найдено по формуле: $\beta_i = \frac{f_i}{t_i}$, где f_i - количество зафиксированных сбоев, которые произошли на i-ом вычислительном узле за время его работы в составе распределенной системы, t_i - суммарное время, которое i-ый вычислительный узел проработал в составе распределенной системы. Однако эта формула обладает тем недостатком, что узел должен проработать достаточно долгое время, прежде чем можно полагаться на значение, полученное по этой формуле. Так до возникновения первого сбоя на этом вычислительном узле его надежность будет равна 1. А это, как правило, далеко от реальности и может значительно исказить получаемые результаты. В худшем случае этот узел может обвинить в своей ошибке другой надежный узел, и т.к. надежность первого узла будет абсолютна, то виновным будет признан второй вычислительный узел. Поэтому предлагается использовать следующую формулу:

$$\beta_i = \frac{1+f_i}{1/\beta + t_i} \quad (1)$$

где β - вероятность отказа вычислительного узла, о котором еще не собрано никакой статистики.

Значение β можно находить по формуле:

$$\beta = \frac{1+\sum_{i=1}^{m} f_i}{\frac{1}{\overline{\beta}} + \sum_{i=1}^{m} t_i} \quad (2)$$

где m – общее количество вычислительных узлов в распределенной системе (не следует путать с n – количеством вычислительных узлов в группе, выделенной специально для решения конкретной задачи)

$\overline{\beta}$ - задаваемая создателем распределенной системы на основе его опыта среднестатистическая вероятность отказа вычислительного узла, которая необходима в начальный момент работы распределенной системы, когда еще не собрано никакой статистической информации.

Формула (1) обладает тем свойством, что при первоначальном подключении вычислительного узла к распределенной системе, когда о нем еще не собрано никакой статистики, значение β_i будет равно $\overline{\beta}$ - вероятности отказа среднестатистического вычислительного узла. При сборе же статистической информации в течение достаточно долгого времени вклад слагаемого $1/\beta$ в знаменатель и 1 в числитель будет пренебрежимо мал: $\beta_i = \lim_{t_i \to \infty} \frac{1+f_i}{1/\beta + t_i} = \lim_{t_i \to \infty} \frac{f_i}{t_i}$.

Аналогичное замечание относиться и к формуле (2). В момент начала работы распределенной системы, когда не собрано никакой статистики, вероятность отказа среднестатистического вычислительного узла будет определяться на основе опыта создателя системы. При накоплении же статистики вклад этого слагаемого будет неуклонно уменьшаться и заменяться эмпирическими данными, полученными самой распределенной системой.

4. **Формула и эффективный алгоритм вычисления вероятности сбоя отказоустойчивой системы**

Имея данные о вероятности отказа каждого вычислительного узла в отдельности, можно рассчитать вероятность сбоя отказоустойчивого решения, объединяющего несколько вычислительных узлов.

Будем считать, что отказы вычислительных узлов – независимые события, т.к. именно к этому стремиться подсистема балансировки при создании групп и распределении нагрузки. Тогда вероятность того, что в наборе из n вычислительных узлов произойдет ровно k отказов вычисляется по формуле:

$$\alpha_n^k = \sum_{F \subseteq U, |F|=k} \left(\prod_{i \in F} \beta_i\right)\left(\prod_{j \in U \setminus F} (1-\beta_j)\right) \quad (3)$$

где F – неупорядоченное подмножество из множества всех вычислительных узлов – U.

Смысл формулы (3) прост – суммируются вероятности возникновения всех возможных комбинации из k выходящих из строя вычислительных узлов, и n-k узлов продолжающих корректно работать.

Так как все приведенные выше отказоустойчивые системы гарантируют корректную работу лишь при условии, что не более чем $f = \left[\frac{n-1}{3}\right]$ вычислительных узлов выйдут из строя, то при возникновении более чем f сбоев вся система целиком может дать сбой. В работе [5] доказано, что ни один алгоритм не может обеспечить корректную работу системы из n вычислительных узлов более чем при $\left[\frac{n-1}{3}\right]$ отказах. Там же строится пример, когда $\left[\frac{n-1}{3}\right]+1$ отказов приводят к сбою системы целиком. А так как рассматривается Византийская модель сбоев, когда вышедшие из строя узлы могут находиться под *согласованным* управлением злоумышленников, то необходимо рассматривать худший случай и считать, что при отказе более f узлов вся система дает сбой. Тогда вероятность этого можно найти по формуле:

$$\alpha_n = \sum_{k=\left[\frac{n-1}{3}\right]+1}^{n} \alpha_n^k = \sum_{k=\left[\frac{n-1}{3}\right]+1}^{n} \sum_{F \subseteq U, |F|=k} \left(\prod_{i \in F} \beta_i\right)\left(\prod_{j \in U \setminus F} (1-\beta_j)\right) \quad (4)$$

Можно воспользоваться тем фактом, что $\sum_{k=0}^{n} \sum_{F \subseteq U, |F|=k} \left(\prod_{i \in F} \beta_i\right)\left(\prod_{j \in U \setminus F} (1-\beta_j)\right)$ (т.е. сумма вероятностей того, что откажет от нуля до n узлов равна 1, т.к. это событие покрывает все возможные исходы). Тогда получим формулу, требующую суммирования меньшего числа слагаемых:

$$\alpha = 1 - \sum_{k=0}^{\left[\frac{n-1}{3}\right]} \sum_{F \subseteq U, |F|=k} \left(\prod_{i \in F} \beta_i\right)\left(\prod_{j \in U \setminus F} (1-\beta_j)\right) \quad (5)$$

Выражение

$$\sum_{k=0}^{\left[\frac{n-1}{3}\right]} \sum_{F \subseteq U, |F|=k} \left(\prod_{i \in F} \beta_i\right)\left(\prod_{j \in U \setminus F} (1-\beta_j)\right) \quad (6)$$

представляет вероятность того, что из строя выйдет не более чем $\left[\frac{n-1}{3}\right]$ вычислительных узлов (т.е. существующие отказоустойчивые решения могут гарантировать корректную работу системы в целом), поэтому его можно назвать надежностью полученного решения.

В случае, если вероятности отказа всех узлов примерно одинаковы и равны β, формулы (4) и (5) можно упростить, воспользовавшись тем, что:

$$\left(\prod_{i \in F} \beta_i\right)\left(\prod_{j \in U \setminus F} (1-\beta_j)\right) = \beta^{|F|} * (1-\beta)^{n-|F|}$$

и количество всевозможных комбинаций выбора k неупорядоченных элементов из множества мощностью n равно C_n^k получим:

$$\alpha = 1 - \sum_{k=0}^{\left[\frac{n-1}{3}\right]} C_n^k \beta^k * (1-\beta)^{n-k} \quad (7)$$ – аналог формулы (5) для случая, когда вероятности сбоя вычислительных узлов примерно одинаковы. Хотя эта формула и является частным случаем (5), но с практической точки зрения удобна тем, что требуется гораздо меньше операций для ее вычисления.

Применение формулы (5) “в лоб” на практике крайне не желательно, т.к. она требует перебора всех возможных подмножеств мощностью от 0 до $\left[\frac{n-1}{3}\right]$ множества из n элементов. Что требует порядка

$\sum_{k=0}^{\left[\frac{n-1}{3}\right]} C_n^k$ операций. Кроме того, полученные подмножества необходимо еще перемножать, тогда получаем $n * \sum_{k=0}^{\left[\frac{n-1}{3}\right]} C_n^k$ операций. А это выражение растет экспоненциально с ростом n.

Для получения эффективного способа вычисления общей надежности вернемся к формуле (3) и разобьем сумму, входящую в ее состав на две части. Первая часть, где n-ый вычислительный узел входит во множество отказавших узлов, а во второй, где он продолжает работать без сбоев. Получим:

$$\alpha_n^k = \sum_{F \subseteq U, |F|=k, n \in F} (\prod_{i \in F} \beta_i)(\prod_{j \in U \setminus F} (1-\beta_j)) + \sum_{F \subseteq U, |F|=k, n \notin F} (\prod_{i \in F} \beta_i)(\prod_{j \in U \setminus F} (1-\beta_j))$$

Вынесем множители с β_n за знаки сумм:

$$\alpha_n^k = \beta_n \sum_{F \subseteq U \setminus \{n\}, |F|=k-1} (\prod_{i \in F} \beta_i)(\prod_{j \in U \setminus \{n\} \setminus F} (1-\beta_j)) + (1-\beta_n) \sum_{F \subseteq U \setminus \{n\}, |F|=k} (\prod_{i \in F} \beta_i)(\prod_{j \in U \setminus \{n\} \setminus F} (1-\beta_j))$$

Введем обозначение $U' = U \setminus \{n\}$, и рассмотрим выражение:

$$\sum_{F \subseteq U', |F|=k-1} (\prod_{i \in F} \beta_i)(\prod_{j \in U' \setminus F} (1-\beta_j))$$

оно представляет собой ни что иное как вероятность того, что ровно k-1 узел выйдет из строя в наборе U', состоящем из n-1 вычислительных узлов, и равняется α_{n-1}^{k-1}.

Аналогично:

$$\sum_{F \subseteq U', |F|=k} (\prod_{i \in F} \beta_i)(\prod_{j \in U' \setminus F} (1-\beta_j))$$

Получили рекурсивную формулу:

$$\alpha_i^k = \beta_i \alpha_{i-1}^{k-1} + (1-\beta_i)\alpha_{i-1}^k \quad (8)$$

Базой рекурсии будут служить выражения:

$\alpha_i^0 = \prod_{j=1}^{i} (1-\beta_j) = \alpha_{i-1}^0 (1-\beta_i)$, при i>0. Вероятность того, что не будет сбоев ни на одном из i узлов.

$\alpha_i^i = \prod_{j=1}^{i} \beta_j = \alpha_{i-1}^{i-1} \beta_i$, при i>0. Вероятность того, что сбои произойдут на всех i узлах.

$\alpha_0^0 = 1$

Применение формулы (8) напрямую опять будет сопряжено с экспоненциальным по отношению к n ростом затрат вычислительных мощностей. Однако можно заметить, что большинство значений α_i^k будет вычисляться по нескольку раз, и если однажды вычисленные значения хранить в таблице, то сложность алгоритма будет $O(n^2)$. Это следует из того, что размер таблицы не будет превышать (n+1)x(n+1). А на вычисление значения одной ячейки таблицы будет требоваться максимум две операции умножения, и по одной сложения и вычитания действительных чисел.

Текст алгоритма на псевдокоде:

Объявить двумерный массив A, в котором будут храниться значения α_i^k

Объявить массив b[i] для хранения данных о надежности отдельных вычислительных узлов

```
Функция a(k,i)
      Если A[k,i] == -1, то
            Если k = 0, то A[k,i] = a(0, i-1) * (1 - b[i])
            Иначе, если k = i, то A[k,i] = a(k - 1, i - 1) * b[i]
                  Иначе A[k,i] = a(k - 1, i - 1) * b[i] + a(k, i - 1) * (1
- b[i])
      Вернуть A[k,i]
Конец функции a.
```

Инициализировать элементы массива b по формуле (1)

Инициализировать элементы массива A[n,n] значениями -1
```
A[0,0] = 1
Надежность = a(0, n) + a(1, n) + … + a((n-1)/3, n)
Вероятность отказа = 1 - Надежность
```

5. **Оценка надежности существующих систем при использовании неоднородных по надежности компонент**

Имея формулу и алгоритм для оценки надежности отказоустойчивых решений, рассмотрим несколько примеров, в которых применение существующих систем приводит к плохим результатам.

Пример 1:

Рассмотрим случай, когда имеется один высоконадежный (вероятность сбоя равна 0.00001) сервер. Единственный способ повышения надежности, доступный существующим системам, заключается в увеличении параметра f – максимально допустимом количестве вычислительных узлов, выход которых из строя может выдержать система. В данном случае пусть f изменится с 0 (система не выдержит ни одного отказа, но благодаря высокой надежности сервера вероятность его отказа очень мала) на 1. Это требует увеличения количества используемых вычислительных узлов с 1 до 4, т.е. необходимо 3 дополнительных вычислительных узла. Если вероятность сбоя добавляемых узлов велика (например, 0.01), то надежность получившийся системы будет меньше исходной.

Расчет надежности получившийся системы в соответствии с алгоритмом из предыдущего раздела приводится на рисунке 1. Итоговая надежность будет равна сумме чисел из отмеченных цветом ячеек - 0,99970170597. А вероятность сбоя системы целиком - 0,00029829403. Таким образом, добавив 3 дополнительных узла, получили группу, которая хотя и может выдержать отказ одного из узлов, но все-таки вероятность сбоя двух или более узлов в 30 раз выше, чем если бы использовался один высоконадежный сервер.

i / k	-	$\beta_1 = 0.00001$	$\beta_2 = 0.01$	$\beta_3 = 0.01$	$\beta_3 = 0.01$
	0	1	2	3	4
0	1,00000000000	0,99999000000	0,98999010000	0,98009019900	0,97028929701
1	-1,00000000000	0,00001000000	0,01000980000	0,01980960300	0,02941240896
2	-1,00000000000	-1,00000000000	0,00000010000	0,00010019700	0,00029729106
3	-1,00000000000	-1,00000000000	-1,00000000000	0,00000000100	0,00000100296
4	-1,00000000000	-1,00000000000	-1,00000000000	-1,00000000000	0,00000000001

Рисунок 1. Пример расчета надежности группы, состоящей из 4 узлов.

На приведенный пример можно взглянуть и с другой стороны. Пусть у нас имеется система с четырьмя вычислительными узлами, вероятность сбоя каждого из которых равна 0.01. Тогда вероятность отказа всей системы будет равна 0,00059203. Если теперь заменить один из узлов, на узел с вероятностью отказа в 1000 раз меньше, то выигрыш в вероятности отказа полученной системы будет менее двух крат.

Пример 2:

Пусть в группе будет примерно поровну высоконадежных и ненадежных вычислительных узлов ($\beta_{1,2,3}=0{,}0006$, $\beta_{4,5,6,7}=0.01$). Получим вероятность сбоя всей системы 0,00000506998066.

Стоит отметить следующее: если бы система состояла только из ненадежных вычислительных узлов, то вероятность отказа была бы равна 0,00003396253015, а если бы система состояла только из надежных узлов, то 0,00000000754640. Т.е. вероятность сбоя уменьшается в 4500 раз. Следовательно, можно бы ожидать, что замена одного ненадежного узла на надежный уменьшала бы общую вероятность отказа примерно в 3,326 раза. Тогда можно было бы ожидать, что общая надежность системы из трех надежных и четырех ненадежных узлов была бы равна $0{,}00003396253015 * 3{,}326^3 = 0{,}000000923$, что в 5,5 раз лучше, чем полученная.

Пример 3:

Параметризируем пример 2. Пусть у нас в системе n=10 вычислительных узлов. Введем параметр p – количество надежных (вероятность сбоя 0.001) узлов. Остальные n-p – имеют вероятность отказа 0.01. p изменяется от 0 до n. Полученные результаты зависимости вероятности отказа всей системы от количества надежных вычислительных узлов представлены на рисунке 2.

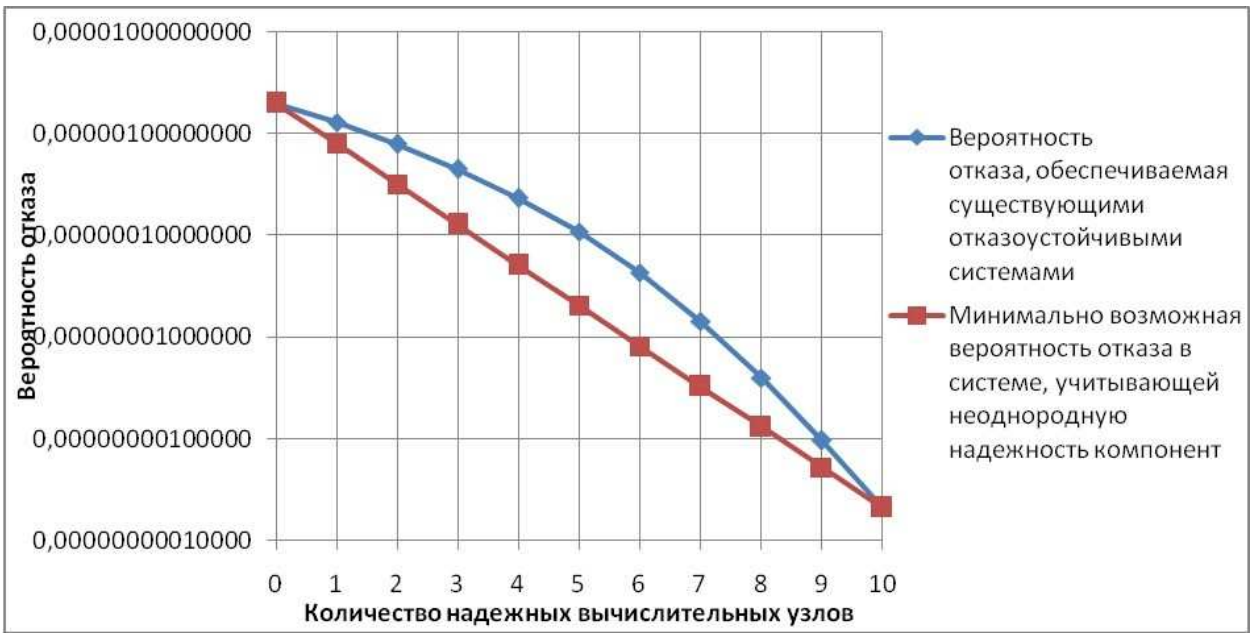

Рис. 2. Иллюстрация зависимости общей надежности системы от количества надежных процессоров

Как видно из графика, в существующих отказоустойчивых системах при доле надежных вычислительных узлов меньше половины, отдача от замены ненадежных вычислительных узлов на надежные значительно меньше (следует обратить внимание на логарифмическую шкалу), чем в случае, когда доля надежных узлов больше половины, хотя абсолютные значения вероятности сбоя в обоих случаях одинаковы.

6. **Повышение надежности за счет группировки вычислительных узлов.**

Из предыдущих разделов следует, что чем больше различие в надежности вычислительных узлов, тем менее эффективны существующие алгоритмы. И наоборот если разница в надежности вычислительных узлов минимальна, то алогоритмы работают максимально эффективно. Поэтому предлагается следующий способ повышения надежности существующих алгоритмов: применять их не ко всей группе целиком, т.к. группа может быть очень разнородна (а следовательно алгоритм не эффективен), а выделить в группе некоторую подгруппу вычислительных узлов близких по надежности. Подгруппа вычислительных узлов, взаимодействующих по отказоустойчивому алгоритму, представляет собой «виртуальный» вычислительный узел с более высокой надежностью, чем любой из составляющих его реальных вычислительных узлов. В свою очередь полученный «виртуальный» вычислительный узел может быть включен в другие подгруппы с близкими по надежности виртуальными или реальными вычислительными узлами, образуя новые виртуальные узлы. Полученный в итоге виртуальный вычислительный узел с максимальной отказоустойчивостью будет обладать как минимум не меньшей надежностью, чем решение без группировки (т.к. подгруппа может быть только одна и включать в себя все вычислительные узлы). А положительный эффект от группировки будет тем больше, чем больше неоднородность вычислительных узлов.

Алгоритм нахождения оптимального способа группировки вычислительных узлов:

1. На вход алгоритму подается U – множество доступных вычислительных узлов. Каждый вычислительный узел характеризуется вероятностью отказа β_i .
2. Если множество U состоит из одного элемента, то этот элемент возвращается в качестве результата.
3. Перебираются всевозможные подмножества G множества U, мощность которых больше 1. $G \subseteq U, |G| > 1$

a) Для каждого G с использованием алгоритма из 5ого раздела вычисляется вероятность отказа группы целиком β_G .

b) Формируется новое множество вычислительных узлов, из которого исключены все узлы, входящие в G, и добавлен новый «виртуальный» вычислительный узел {G}. $U' = (U \setminus G) \cup \{\beta_G\}$

c) Алгоритм вызывается рекурсивно для множества U'.

d) Из всех возможных вариантов выбора множества G, используется тот, при котором достигается минимум вероятности отказа.

Для сокращения количества вариантов перебора на множество G следует накладывать следующие ограничения:

1. $|G| = 3k + 1, k \in N$
2. G должна объединять близкие по надежности вычислительные узлы.

На рисунке 3 продемонстрировано работа алгоритма на входных данных из второго примера предыдущего раздела. Как видно, группировка позволяет повысить надежность системы в 2,5 раза.

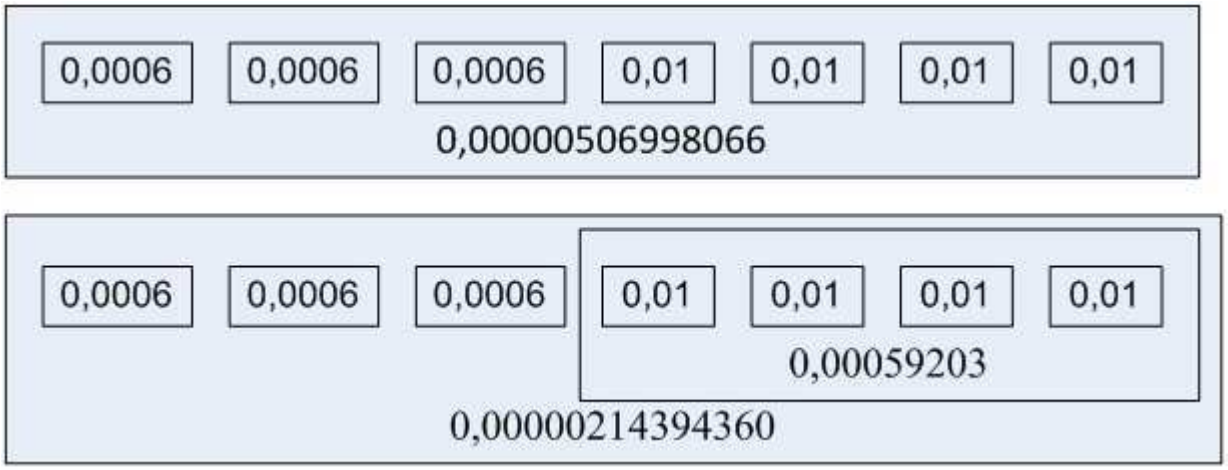

Рис. 3. Иллюстрация уменьшения вероятности отказа за счет группировки вычислительных узлов

7. Заключение

В статье было продемонстрировано, как сбор статистических данных о надежности вычислительных узлов помогает в обеспечении более высокой отказоустойчивости создаваемых решений, особенно для гетерогенных распределенных систем. Это далеко не единственный пример работы более общего подхода, основная идея которого заключается в том, что чем более полная информация имеется о среде, в которой работает система, то тем более эффективное решение можно подобрать для этой конкретной ситуации [6-8]. Общие же решения зачастую оказываются гораздо менее эффективными, особенно если среда не однородна.

ЛИТЕРАТУРА:

1. Lamport L., Shostak R. and Pease M. The Byzantine Generals Problem. // ACM Transactions on Programming Languages and Systems –July 1982. –Vol. 4, N.3. –P. 382-401.
2. Reiter M. The Rampart Toolkit for Building High-Integrity Services // Theory and Practice in Distributed Systems –Springer-Verlag, 1995.
3. Kihlstrom K. P., Moser, L. E. and Mellar-Smith, P. M. The SecureRing group communication system // ACM Transactions on Information and System Security (TISSEC), –November 2001. –Vol. 4, N. 4. –P. 371-406.
4. Castro M. Practical Byzantine Fault Tolerance // Massachusetts Institute of Technology, 2001. PhD thesis.
5. Bracha G. and Toueg S. Asynchronous Consensus and Broadcast Protocols // Journal of the ACM. –October 1985. –Vol. 32, N.4. –P. 824-840.
6. Фирсов А.Н. Оптимизация на основе статистических данных асинхронной распределенной системы, устойчивой к произвольным отказам // Параллельные вычислительные технологии (ПаВТ’2009): Труды международной научной конференции (Нижний Новгород, 30 марта – 3 апреля 2009 г.). – Челябинск: Изд. ЮУрГУ, 2009г. – С.765-771.
7. Фирсов А.Н. Оценка эффективности некоторых оптимизаций протоколов надежной и атомарной групповой рассылки // Вестник Пермского госуниверситета. Математика. Механика. Информатика. 2009г. – С.161-168.
8. Фирсов А.Н. Статистически оптимизируемая отказоустойчивая распределенная система // III конференция-конкурс грантов аспирантов и молодых ученых механико-математического факультета Пермского государственного университета. Сборник тезисов научных докладов конференции – Пермь, 2008г. – С.74-78.