АНАЛИЗ ОТКАЗОУТОЙЧИВОСТИ ВЫЧИСЛИТЕЛЬНОЙ СРЕДЫ ПЛАНЕТАРНОГО ТИПА

Коганов А. В., Сазонов А. Н.

(Россия, Москва)

Рассматриваются модели вычислительных сред (ВС), построенные в форме направленных графов с мечеными вершинами. Моделью задачи для ВС называется некоторый подграф этой среды. Под разрушением ВС понимается выделение некоторого собственного подграфа модели (РВС). Приводится оценка сложности задачи поиска образа графа задачи на графе ВС. Методом имитационного моделирования исследуется зависимость времени отказа (невозможности нахождения образа задачи) от параметров графа ВС для специального класса ВС – планетарных вычислительных систем.

1. Введение. Общая проблематика анализа отказоустойчивости, а также основные понятия приведены в работе «Анализ отказоустойчивости вычислительной среды корпоративного типа» в этом же сборнике. Поэтому здесь мы не будем останавливаться на определениях модели разрушенной вычислительно системы, пакета задач и других. В упомянутой работе приводятся аналитические выкладки для моделей вычислительных систем корпоративного типа. В данной работе рассмотрены вопросы сложности поиска образа задачи на графе РВС, а также описаны реализованные авторами алгоритмы поиска переназначений пакета задач как для случая общей вычислительной среды, так и для важного частного случая сети планетарного типа, где возможно ускоренное решение задачи. Проведено описание и результаты численного статистического анализа отказоустойчивости таких сетей в зависимости от нескольких параметров.

2. Общий алгоритм перераспределения. Перераспределение задачи на РВС фактически является задачей поиска подграфа графа РВС, изоморфного графу задачи. Для нераскрашенных графов доказано, что такая задача является NP-полной [3]. Нераскрашенный граф является частным случаем раскрашенного (одноцветного), и, таким образом, задача для раскрашенных графов также является NP-полной.

Обращаясь к природе узла графа РВС как узла ВС, можно немного модифицировать задачу. Предположим, в графе задачи есть несколько соседних одноцветных вершин, последовательно связанных друг с другом. В этом случае можно распределять эти вершины на одну и ту же вершину графа РВС, предполагая, что локальные пересылки данных (на одном и том же узле РВС) всегда возможны. Поэтому образом нескольких вершин графа задачи может быть одна и та же вершина графа РВС. Такую ситуацию будем называть *склейкой вершин графа задачи* или просто *склейкой*.

2.1. Описание алгоритма поиска подграфа РВС, изоморфного графу задачи. Пусть графы задачи и РВС связные. Иначе разобьем их на связные подграфы и будем применять алгоритм для каждой пары полученных подграфов в отдельности.

Постановка задачи. Множество вершин произвольного графа G обозначим N(G), а множество его ребер – L(G).

Пусть имеются два связных, раскрашенных направленных графа S и D, где S — граф задачи, D — граф РВС.

Для любой вершины $s \in N(S)$ и для любой вершины $d \in N(D)$ определен цвет (тип) c(s) и c(d) соответственно из множества цветов C.

Требуется построить соответствие вершин графов R такое, что для любой вершины $s \in N(S)$ $R(s)=d \in N(D)$ и верны следующие утверждения:

A1) c(s)=c(d).

A2) Для любого ребра $l_s \in L(S)$, направленного от $s_1 \in N(S)$ к $s_2 \in N(S)$ существует ребро $l_d \in L(D)$, направленное от $d_1 = R(s_1) \in N(D)$ к $d_2 = R(s_2) \in N(D)$.

Замечание 2.1. Из (A2) следует, что для любых индексов $k_1 \ldots k_m$, таких что $k_i <= |N(S)|$ и существуют дуги между s_{ki} и $s_{k(i+1)}$ при $i=1 \ldots (m-1)$, также существуют дуги графа D между $R(s_{ki})$ и $R(s_{k(i+1)})$, направленные одинаково (в сторону увеличения или уменьшения i соответственно).

Вариант алгоритма.

1. Для каждой вершины $s_i \in N(S)$ построим множество $V_i = V(s_i) \subseteq N(D)$ такое, что для любой вершины $d_i \in V_i$ верно, что:
   1.1. $c(s_i) = c(d_i)$.
   1.2. Для любой вершины $s \in N(S)$ такой, что существует ребро $l_{si} \in L(S)$, направленное от s к s_i, существует вершина $d \in N(D)$ такая, что существует ребро $l_{di} \in L(D)$, направленное от d к d_i и $c(d) = c(s)$.
   1.3. Для любой вершины $s \in N(S)$ такой, что существует ребро $l_{si} \in L(S)$, направленное от s_i к s, существует вершина $d \in N(D)$ такая, что существует ребро $l_{di} \in L(D)$, направленное от d_i к d и $c(d) = c(s)$.
      1.3.1. Такие множества V(si) будем называть множеством вариантов для вершины si.
2. Отсортируем вершины $s_i \in N(S)$ по количеству вариантов (по $|V_i|$), начиная с меньшего значения. Иногда это ускоряет обнаружение невозможности переназначения.
3. Положим n — глубину рекурсии, совпадающую с номером анализируемой вершины, равной единице.
4. Положим номер анализируемого варианта $k_n = 1$.
5. Положим $R(s_n) = d_{kn} \in V_n$.
6. Запустим процедуру частичной проверки, которая определяет корректность отображения $R(s_n)$ для всех предыдущих вершин (описание дано ниже).

6.1. Если процедура проверки завершилась удачно, то:
  6.1.1. Если n=|N(S)| получен требуемый результат, переход на шаг 7.
  6.1.2. Иначе увеличиваем n на единицу и запускаем алгоритм с шага 4. Если алгоритм не нашел решения и «вернулся», то, если $k_n < |V_n|$, увеличиваем k_n на единицу и переходим на шаг 5, иначе возвращаемся на предыдущую итерацию (уменьшаем n на единицу). Если n стало равным нулю, то решения не существует.

6.2. Если процедура проверки завершилась неудачно, то, если $k_n < |V_n|$, увеличиваем k_n на единицу и переходим на шаг 5, иначе возвращаемся на предыдущую итерацию (уменьшаем n на единицу). Если n стало равным нулю, то решения не существует.

7. Набор $R(s_n)=d_{kn}$, где n =1…|N(S)|, – требуемый результат.

Процедура проверки. Для процедуры проверки входными данными являются графы S и D, а также набор $R(s_i)$, где i=1…n, n — текущая глубина рекурсии. Алгоритм проверки.

Для каждой вершины s_i, где i=1…n, определим множество дуг $L_{si} \subseteq L(S)$, исходящих из вершины s_i,и множество дуг $L_{di} \subseteq L(S)$, входящих в вершину s_i. Для каждой дуги $l_{si} \in L_{si}$ найдем вершину s_j, в которую идет дуга l_{si}. Если j>n, т.е. еще не определено $R(s_j)=d_{kj}$, то пропустим эту дугу. Иначе проверим, что в графе D существует дуга, исходящая из $d_{ki}=R(s_i)$ и идущая в $d_{kj}=R(s_j)$. В случае если вершины d_{ki} и d_{kj} совпадают, будем считать, что дуга существует. Если такой дуги нет, то процедура проверки дает отрицательный результат. Иначе проверяем остальные дуги. Аналогично проверяем и дуги $l_{di} \in L_{di}$. В случае если все дуги существуют, процедура проверки дает положительный результат.

Если n<|N(S)|, то $R(s_i)$ – частичное решение, иначе $R(s_i)$ – полное решение.

Утверждение 2.1 Для любой вершины $s_i \in N(S)$ и для любого отображения R, являющегося решением задачи, верно, что $R(s_i) \in Vi$.

Доказательство. Пусть существует отображение R, являющееся решением задачи такое, что существует $R(s_i)$, не принадлежащее V_i. Тогда по построению множества V_i либо

1) $c(s_i) \neq c(R(s_i))$, либо

2) существует некоторый цвет B и такая вершина $s_j \in N(S)$, что $c(s_j)=B$ и существует дуга из s_j в s_i (аналогично для дуги из s_i в s_j), причем не существует вершины d_k, такой что $c(d_k)=B$, и существует дуга из d_k в $R(s_i)$ (аналогично для дуги из $R(s_i)$ в d_k).

В первом случае нарушено условие задачи A1 и преобразование R не является решением.

Во втором случае не существует дуги из (в) $R(s_j)$ в (из) $R(s_i)$, что противоречит условию задачи A2

Такой алгоритм решения задачи допускает решения со склейками. Для доказательства NP-полноты задачи со склейками требуется доказать ее полиномиальную сводимость к задаче поиска подграфа, изоморфного данному.

2.2. Сведение задачи со склейками к задаче поиска изоморфного раскрашенного подграфа

Постановка задачи. Пусть даны два направленных графа S $N(S)=V_s$, $L(S)=E_s$ и D $N(D)=V_d$, $L(D)=E_d$, вершины которых раскрашены в цвета из множества C. Пусть также дан алгоритм R, любой вершине $v_s \in Vs$ ставящий в соответствие вершину $v_d \in V_d$ (будем называть ее образом v_s) таким образом, что выполнены условия A1 и A2:

Теорема 2.1. Возможно дополнить графы S, D новыми вершинами и ребрами, а также расширить множество цветов вершин C так, чтобы алгоритм R при отображении на модифицированных графах не допускал склеек вершин и при этом находил для исходных графов отображение без склеек, если оно существует.

Доказательство. Рассмотрим конструкцию. Для каждого цвета c_i из C добавим новый цвет c_{i1}. Полученное множество цве-

тов назовем C_r. Разобьем множества вершин V_s и V_d на множества V_{si} и V_{di} (i=1…|C|) так, что в каждом из множеств V_{si} и V_{di} будут все вершины цвета c_i. Для каждой пары вершин v_{s1} и $v_{s2} \in V_{si}$ добавим в V_s две вершины цвета c_{i1} (x_{i1} и x_{i2}). Добавим в множество E_s дуги от вершины v_{s1} к x_{i1}, от x_{i1} к v_{s2}, от v_{s2} к x_{i2}, от x_{i2} к v_{s1}. Дополненные множества V_s и E_s назовем V_{sr} и E_{sr} и получим граф $S_r(V_{sr},E_{sr})$. Аналогично дополним граф D и получим граф $D_r(V_{dr},E_{dr})$.

Лемма 2.1.

1) Для графов S_r и D_r при применении алгоритма R склейка невозможна.

2) Для любого соответствия вершин графов S и D, полученного в результате работы алгоритма R, не имеющего склеек, существует и соответствие графов S_r и D_r, удовлетворяющее условиям работы алгоритма, и такое, что для любой вершины s_i из V_s с образом d_i из V_d образ s_{ir} (вершины из V_{sr}, соответствующей s_i) будет вершиной d_{ir} из V_{dr}, соответствующей той же самой вершине d_i.

3) Любое отображение S_r на D_r алгоритмом R включает в себя отображение подграфа S на подграф D этим же алгоритмом.

Доказательство.

1) Пусть при применении алгоритма R к модифицированным графам S_r и D_r была получена склейка вершин s_m и s_k с образом d_i. Так как склейка возможна только для вершин одного цвета, по построению графа S_r между вершинами s_m и s_k существует путь s_m->s_x->s_k, где вершина s_x имеет цвет c_x, добавленный в множество цветов при модификации графов. Тогда по условиям работы алгоритма R должен существовать путь d_i->d_x->d_i, где вершина d_x имеет тот же цвет c_x. Но в графе D_r нет петель, проходящих только через вершину цвета c_x, т.к. этот цвет был добавлен при модификации. Следовательно, при применении алгоритма R к модифицированным графам S_r и D_r склейка невозможна. (1)

2) Возьмем любое соответствие для исходных графов S и D, допустимое по условиям работы алгоритма R, и обозначим его P.

Образами любых двух вершин одного цвета из графа S по условию отсутствия склеек будут разные вершины графа D. Добавим соответствие добавленных вершин между всеми образами вершин графа S одного цвета в соответствии с порядком добавления дополнительных вершин. Такое соответствие всегда найдется, т.к. были добавлены одинаковые и симметричные структуры вершин и ребер. Причем среди добавленных вершин также не будет склеек, т.к. среди оригинальных вершин склеек нет, а значит, и все дополнительные вершины графа D_r являются образами не более чем одной дополнительной вершины графа S_r. (2)

3) Возьмем любое соответствие для графов S_r и D_r, полученное в результате применения алгоритма R, и отбросим все вершины и дуги, добавленные при модификации графов (из S в S_r и из D в D_r). Полученные графы будут S и D, т.к. мы применили обратную модификацию графов. Из полученного в результате работы алгоритма R соответствия P_r исключим соответствия удаленных вершин. Полученное соответствие обозначим P. P будет соответствием для графов S и D, удовлетворяющим условиям работы алгоритма R, т.к.:

3.1) цвета образов и их вершин совпадут по условиям, выполненным для P_r;

3.2) для любых вершин v_{s1} и $v_{s2} \in V_s$, инцендентных ребру $e_s \in E_s$, будет верно, что их образы v_{d1} и v_{d2} инцендентны дуге e_d, и направление дуги e_d будет совпадать с направлением дуги e_s в силу выполнения этих условий для P_r и того, что из S_r и D_r не было удалено ни одной вершины и ни одного ребра, принадлежащих графам S или D соответственно.

(3) Из леммы 2.1 следует теорема 2.1.

Следствие. Максимальное число добавленных вершин в графе G_r будет достигаться в том случае, когда все вершины оригинального графа G имеют один цвет. Тогда $|N(G_r)|=|N(G)|^2$. Таким образом, задача со склейками полиномиально сводима к задаче поиска подграфа раскрашенного графа, изоморфного данному, и, следовательно, является NP-полной как в случае с возможностью склеек, так и в случае отсутствия склеек.

Были реализованы модификации алгоритма путем варьирования глубины поиска вариантов для каждой вершины до двух звеньев цепи, начала и частоты проверки частичного решения, а также способа сортировки вершин (например, по кратности).

Также реализован алгоритм, который оптимизирует количество склеек, но это может привести к существенному замедлению поиска решения.

3. Исследование отказоустойчивости для МВС планетарного типа

3.1 Определение МВС планетарного типа. В связи с NP-полнотой общей задачи нахождения подграфа, изоморфного данному, а также с неоднозначностью оценки сложности произвольного графа для исследования отказоустойчивости были выбраны ВС планетарного типа.

Граф планетарных вычислительных систем (ПВС) двунаправленный и состоит из центральных элементов (звезд), с каждой из которых связаны серверные элементы разных типов (планеты). Планетарная система — подграф ПВС, состоящий из одной звезды и всех ее планет. Звезды между собой связаны полным графом. Цвета всех звезд одинаковы. Цвета планет могут отличаться, однако никогда не совпадают с цветом звезд. Для ПВС был задан процесс генерации вероятностного типа с тремя основными характеристиками:

1. S – количество звезд.
2. C – количество цветов планет.
3. K – среднее количество планет каждого цвета у каждой звезды.

Число планет у звезды равно KC. При генерации ПВС каждая планета получает цвет как реализацию равновероятного испытания Бернулли на множестве цветов 1,…,c.

В качестве пакета задач для исследования отказоустойчивости используется множество всех планетарных систем ПВС.

Разрушение ПВС проводится испытанием Бернулли, путем удаления из графа каждой вершин и инцендентных ей ребер с не-

которой вероятностью Pd. Каждая планета, а также звезда удаляется из графа в случае, если случайный эксперимент с вероятностью успеха Pd окончился успехом.

Разложением одной планетарной системы W1 на другую планетарную систему W2 называется преобразование W1 в W2, при котором звезда отображается на звезду и каждой планете w1 из W1 ставится в соответствие планета w2 из W2 того же цвета (w2 называется образом w1) . Разложение считается успешным, если для всех планет wi1 из W1 существует образ wj2 из W2.

По набору параметров S, C, K проводится анализ отказоустойчивости ПВС.

3.2 Статистическое исследование отказоустойчивости ПВС

3.2.1 Однократный эксперимент. Для данного ПВС задается число итераций разрушения Id. На каждой итерации, после разрушения, производятся попытки разложить все исходные (неразрушенные) планетарные системы по-отдельности на оставшиеся неполностью разрушенные планетарные системы полученного графа. В случае если все исходные планетарные системы удалось разложить на какие-либо планетарные системы из разрушенного графа, итерация считается успешной и алгоритм переходит на следующую итерацию. Иначе номер текущей итерации T называется временем отказа системы для этого эксперимента. На каждой итерации t запоминается количество склеек звезд: для каждой звезды графа ПВС вычисляется величина $Ngs(t) = (Nt(t) - 1)Nt(t)/2$, где Nt(t) — количество звезд-задач, распределенных на данную звезду на данной итерации. Ngs(t) – число фактически произведенных склеек пар вершин на данной итерации для звезды s. Количество склеек пар вершин на данной итерации Ng(t) вычисляется как сумма Ngs(t) по всем звездам. Общее число склеек Ng вычисляется как сумма Ng(t) по всем итерациям t=1…T.

В случае если в процессе разрушения на очередной итерации в графе не осталось ни одной звезды, результат эксперимента помечается как «полное разрушение».

В случае если отказ системы не был получен за Id итераций, результат эксперимента помечается как «Полный успех», время отказа устанавливается равным Id=T.

После окончания всех итераций вычисляется Mg=Ng/T и стандартное отклонение Dg числа склеек по всем итерациям:

$$Dg = \sqrt{\frac{\sum_{t=1...T} (Ng(t) - Mg)^2}{T}}.$$

3.2.2 Многократный эксперимент для одного набора параметров S, C, K

1. Задается параметр Ne – число экспериментов, Id – число итераций разрушения в каждом эксперименте и Pd – вероятность разрушения.
2. Задаются параметры S, C и K.
3. Создаются счетчики D (число полных разрушений) и F (число полных успехов) с начальным значением 0.
4. Для каждого n от 1 до Ne включительно создается реализация G графа ПВС, согласно параметрам S, C и K. Множество реализаций обозначим Q, |Q|=Ne.

4.1. Для созданного графа G проводится эксперимент, описанный в п. 3.2.1, сохраняется полученное время отказа T и количество склеек по итерациям Ng(t) и, возможно, увеличиваются счетчики D или F в случае, если было получено полное разрушение или полный успех соответственно.

Подсчитывается среднее время отказа Md, стандартное отклонение времени отказа Dd, а также среднее количество склеек по всем тестам Mg и стандартное отклонение числа склеек Dg по всем итерациям:

$$Md = \frac{\sum_{G \in Q} T(G)}{Ne} \quad Dd = \sqrt{\frac{\sum_{G \in Q} (T(G) - Md)^2}{Ne}}.$$

3.2.3 Метод исследования зависимости отказоустойчивости ПВС от параметров S, C, K

Для каждого из параметров S, C и K задается минимальное и максимальное значение. Также задаются параметры Id и Pd. Затем для каждого набора значений S, C и K проводится эксперимент, описанный в п. 3.2. Производится анализ полученных данных: построение графика зависимости усредненных значений (Md, Dd, Mg, Dg) по двум из переменных S, C и K от третьей переменной и линейной регрессии указанных средних.

4. Результаты численного эксперимента. Эксперимент проводился с использованием следующих значений параметров:

– вероятность разрушения Pd=0.05;

– итераций разрушения Id=100;

– для каждых S,C,K кратность экспериментов Ne =100.

На рис 1, 2 и 3 приведены результаты моделирования, где при помощи записей вида Md(X), Dd(X), Mg(X), Dg(X), где X — один из параметров S, C, K, обозначены линейные регрессии соответствующих усредненных значений. На основании численного эксперимента можно утверждать следующие тенденции изменения характеристик надежности ПВС:

– зависимость от S, усредненная по C и K: Md, Mg и Dg монотонно возрастают, Dd незначительно убывает;

– зависимость от C, усредненная по S и K: Md, Dd, Mg монотонно убывают, Dg незначительно убывает;

– зависимость от K, усредненная по S и C: Md, Dd, монотонно возрастают, Mg и Dg в целом возрастают.

Результаты экспериментов качественно совпадают с теоретическими из работы «Анализ отказоустойчивости вычислительной среды корпоративного типа», опубликованной в этом же сборнике.

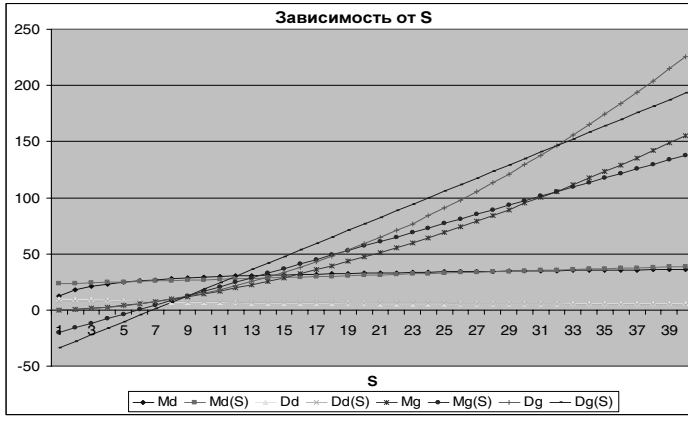

Рис. 1. Зависимость усредненных значений Md, Dd, Mg, Dg по C и K от S

S=1…40, C=1…20, K=1…20
Формулы линейной регрессии:
Md(S)=0.393S + 23.0596; Dd(S)=-0.089S + 8.0952
Mg(S)=4.0401S - 23.8367; Dg(S)=5.8153S - 39.6559

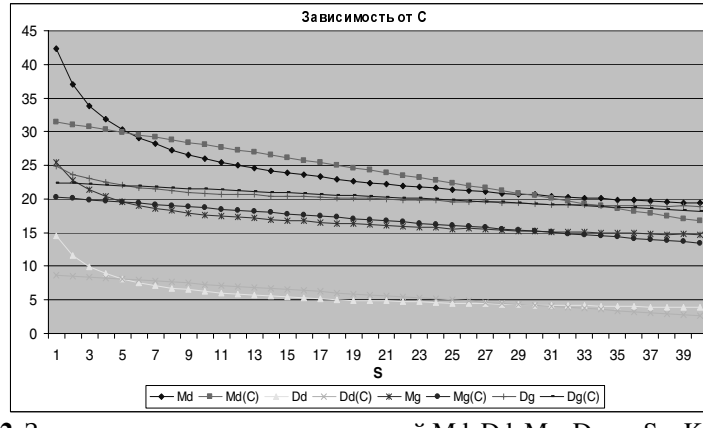

Рис. 2. Зависимость усредненных значений Md, Dd, Mg, Dg по S и K от C

S=1…20, C=1…40, K=1…20

Формулы линейной регрессии:
Md(C)=-0.3778C + 31.8203; Dd(C)=-0.1565C + 8.8905
Mg(C)=-0.1733C + 20.4022; Dg(C)=-0.1075C + 22.5288

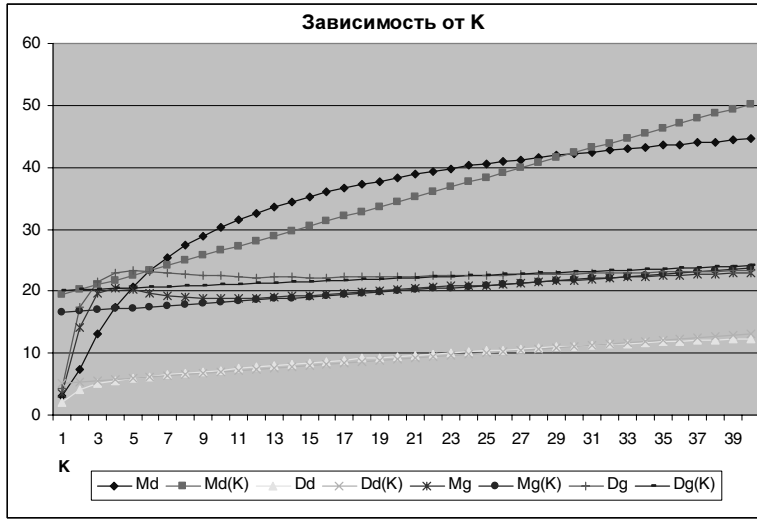

Рис. 3. Зависимость усредненных значений Md, Dd, Mg, Dg по S и C от K

S=1…20, C=1…20, K=1…40
Формулы линейной регрессии:
Md(K)=0.7899K + 18.6447; Dd(K)=0.2081K + 4.8411
Mg(K)=0.1823K + 16.3848; Dg(K)=0.1019K + 19.986

Работа выполнена при поддержке РФФИ, проект № 04-01-00363

СПИСОК ЛИТЕРАТУРЫ

1. Коганов А. В. Индукторные пространства как средство моделирования // Вопросы кибернетики (Алгебра, Гипергеометрия, Вероятность, Моделирование), 1999. — С. 119–181.
2. Крамер Г. Математические методы статистики. — М.: «Мир», 1975.

3. Cook S. A. The complexity of theorem-proving procedures // Annual ACM Symposium on Theory of Computing. 1971. — P. 151–158.

FAULT-TOLERANCE ANALYZIS FOR COMPUTING ENVIRONMENT OF PLANETARY TYPE

Koganow A. V., Sazonov A. N.

(Russia, Moscow)

Computing environment models (CEM), represented by oriented graphs with typed vertices are examined. The model of the task is a sub graph of the graph mentioned. The CEM destruction is its sub graph separation. The complexity of searching for the task's image on the CEM is assessed. Using mathematical simulation the dependencies between the failure time (task's image unavailability) and graph's parameters is examined for special CEM class – Planetary Computing Environment Model.