Задание 3.
Построение сети временных автоматов для решения задачи отказоустойчивого планирования работ в вычислительной системе (UPPAAL) (10 баллов).

Формулировка задания

Дано:

Дан набор программ (далее работ), каждая из которых может быть выполнена на одном из множества процессоров. На набор работ и их выполнение наложены следующие ограничения:

1. Работы при выполнении не могут быть прерваны, а также не мигрируют между процессорами при их выполнении.

2. Между работами есть зависимости по данным. Некоторые работы могут быть готовы к исполнению, после того как другие работы закончены.

3. Каждая работа имеет определенное время исполнения на каждом процессоре. Это означает, что время выполнения на разных процессорах могут быть разными. *Например, представьте себе систему с двумя процессорами: один, который работает с высокой тактовой частотой, но не поддерживает аппаратно инструкции с плавающей запятой (FP), а другой с меньшей частотой, но с аппаратной поддержкой инструкций FP. Некоторые задачи, не содержащие операций FP, выполняются быстрее на первом процессоре. Для других задач FP инструкции эмулируются и таким образом вычисления можно закончить раньше на втором процессоре даже при более низкой тактовой частоте.*

Первый процессор (CPU_1) может иметь резервную копию. Резервная копия может находиться в горячем, тёплом или холодном резерве. Вид резерва или его отсутствие определяется вариантом задания. Подробное описание переключения на резервный компонент можно найти в методических указаниях. Для самой критической из работ может использовать механизм обеспечения отказоустойчивости (МОО). Вид МОО и имя критической работы определяется вариантом задания. Возможны следующие варианты МОО: использование приёмочного теста (AT), N-версионное программирование (NVP_0_1), N-версионное программирование с использованием аппаратного резервирования (NVP_1_1), восстановление блоками (RB_1_1) или контрольные точки (CP). Подробное описание функционирования МОО можно найти в методических указаниях.

Требуется:
1. **(4 балла)** Описать шаблоны автоматов для процессоров и работ. Создать сеть временных автоматов, позволяющих построить расписание выполнения работ на процессорах без учёта резерва процессора и МОО для критической работы.

   Подсказка: необходимо создать два типа параметризованных шаблонов отдельно для процессора, отдельно для работ. Каждый процессор и работа это экземпляр шаблона с заданными параметрами. Взаимодействие между процессорами и работами рекомендуется осуществлять путём посылок сообщений через канал. Также возможно использование глобальных переменных.

2. **(1 балл)** Проверить выполнение следующих свойств:
   - отсутствие тупиков при выполнении модели;
   - каждая задача выполнится, хотя бы на одном процессоре.

В зависимости от варианта задания найти расписание, выполняющееся за минимальное время

(min_time) или с минимальным общим временем простоя процессоров (min_delay). Оценить скорость работы UPPAAL при различных вариантах обхода дерева решений *(search order)*

Подсказка: такое расписание можно найти путём проверки временных свойств и последовательного уменьшения(увеличения) ожидаемого времени.

3. **(3 балла)** Построить модель построения отказоустойчивого расписания дополнив модель, разработанную в п.1. Необходимо промоделировать отказ CPU_1 в момент времени равный 15, а также восстановление работы после отказа. Также необходимо промоделировать работу МОО для критической работы.

4. **(1 балл)** Проверить выполнение следующих свойств:

- отсутствие тупиков при выполнении модели;
- каждая задача выполнится, хотя бы на одном процессоре.

В зависимости от варианта задания найти расписание для модели из п.3, выполняющееся за минимальное время (min_time) или с минимальным общим временем простоя (min_delay) процессоров. Оценить скорость работы UPPAAL при различных вариантах обхода дерева решений *(search order).*

5. **(1 балл)** Описать полученные в п.2 и п.4 результаты в виде отдельного файла. Файл должен содержать результаты для моделей, построенных в п.1 и п.3. Результаты должны включать результаты проверки свойств, описание построенного оптимального расписания, времени его работы или простоя процессоров (в зависимости от варианта), приведены оценки для работы UPPAAL при различных вариантах обхода дерева решений. Также включить в этот файл выводы о возможностях системы UPPAAL и удобстве или неудобстве работы в нём.

На выполнение задания отводится три недели. Последний срок сдачи задания – **17 ноября**.

Требования к файлам решения:

Сданная задача присылается на volkanov@lvk.cs.msu.su письмом с темой Reliability-Task3-Solution. Решение задачи должно включать в себя 6 файлов:

1. Файл с исходными данными year_group_surname_3.csv

2. Файлы year_group_surname_3_1.xml и year_group_surname_3_1.q с моделью и запросами на UPPAAL для модели построения расписания из п.1.

3. Файлы year_group_surname_3_3.xml и year_group_surname_3_3.q с моделью и запросами на UPPAAL для модели построения отказоустойчивого расписания из п.3.

4. Текстовый файл year_group_surname_ucomparision.txt с описанием выводов п.5

Сданная не позднее **17 ноября** и присланная до **23:59 (московское время) 17 ноября** задача принимается (БЕЗ ШТРАФА), далее –3 балла за каждую неделю просрочки.

Методические указания по выполнению задания:

1. **Пример исходных данных**

Пусть дан вот такой файл исходных данных.

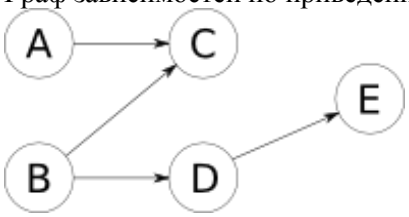
Задача	CPU_1	CPU_2	Зависимости
A	3	4	-
B	3	2	-
C	6	5	A,B
D	3	3	B
E	2	1	D
CPU_1	Горячий_Резерв	1	
C	RB_1_1	1	
Критерий	min_time		

Например, строка

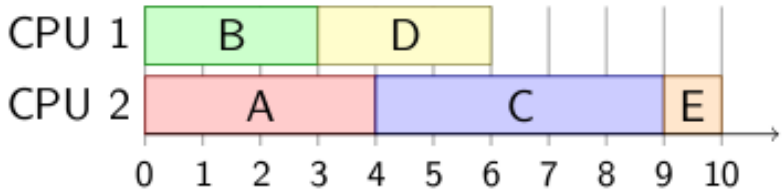
C	6	5	A,B

означает, что задача C выполняется 6 единиц времени на CPU_1 и 5 единиц времени на CPU_2 и не может быть начата, пока не выполнены задачи A,B.

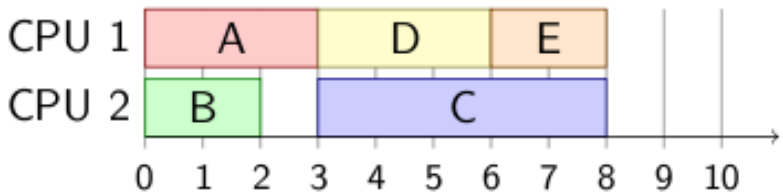
Граф зависимостей по приведённым данным имеет такой вид:



Возможно построение расписания такого вида:



Но оптимальным (для п.1) в случае критерия минимизации времени (min_time) является следующее расписание:



Строка

CPU_1	Горячий_Резерв	1	

означает, что CPU_1 находится в горячем резерве и переключение на резервную копию занимает 1 единицу времени.

Строка

C	RB_1_1	1	

означает, что для критической задачи C используется МОО RB_1_1 и время проведения теста равно 1 единице времени.

Строка

Критерий	min_time		

означает, что в качестве критерия оптимальности расписания является минимизация времени выполнения расписания.

2. Работа в случае отказа процессора CPU_1

В зависимости от варианта задания возможны следующие варианты резерва процессора:

- *отсутствие резерва;*
- *горячий резерв;*
- *тёплый резерв;*
- *холодный резерв.*

Ниже описано как функционирует система после отказа для различных вариантов резерва процессора и список параметров, которые задаются в исходных данных.

2.1 Отсутствие резерва

Параметры:-

В этом случае, если в момент отказа процессора на нём выполнялась работа, то она должна заново запуститься на другом процессоре. После момента отказа на процессоре CPU_1 никакие работы не выполняются.

2.2 Горячий резерв

Параметры:<время_переключения_на_резерв>

В этом случае, если в момент отказа процессора на нём выполнялась работа, то она приостанавливается. В течении заданного параметром времени происходит переключение на резервную копию CPU_1 и работа продолжает своё выполнение с точки останова. После момента отказа работы выполняются на резервной копии процессора CPU_1.

2.3 Тёплый резерв

Параметры:-

В этом случае, если в момент отказа процессора на нём выполнялась работа, то она останавливается. Затем данная работа мгновенно перезапускается с начала на резервной копии CPU_1, то есть работа начинает выполняться с самого начала. После момента отказа работы выполняются на резервной копии процессора CPU_1.

2.4 Холодный резерв

Параметры: <время_переключения_на_резерв>

В этом случае, если в момент отказа процессора на нём выполнялась работа, то она останавливается. В течении заданного параметром времени происходит переключение на резервную копию CPU_1. После переключения работа перезапускается с начала на резервной копии CPU_1, то есть работа начинает выполняться с самого начала. После момента отказа работы выполняются на резервной копии процессора CPU_1.

3. Работа МОО

В зависимости от варианта задания возможны следующие варианты МОО:

- *приёмочный тест (AT);*
- *N-версионное программирование (NVP_0_1);*
- *N-версионное программирование с использованием аппаратного резервирования (NVP_1_1);*
- *восстановление блоками (RB_1_1)*;
- *контрольные точки (CP).*

Ниже описано как функционирует каждый МОО и список параметров, которые задаются в исходных данных.

3.1 Приёмочный тест (AT)

Параметры:<время_тестирования>

Данный МОО включает модуль тестирования результатов работы. В этом случае после выполнения работы запускается приёмочный тест. Время работы теста определяется параметром <время_тестирования>. Если тест не пройден, то результат является ошибочным. При построении расписания считать, что тест всегда отрабатывает успешно.

3.2 N-версионное программирование (NVP_0_1)

Параметры:<время_работы_голосователя>

Данный МОО включает модуль принятия решений (голосователь) и 3 независимо разработанных версий работы. Все 3 версии работают последовательно на одном процессоре, результат их работы обрабатывает голосователь. Время работы голосователя определяется параметром <время_работы_голосователя>. Тот результат, который выдаёт б**о**льшая часть версий, принимается за финальный. Считать, что время работы версий одинаково.

3.3 N-версионное программирование с использованием аппаратного резервирования (NVP_1_1)

Параметры: <время_работы_голосователя>

Данный МОО включает модуль принятия решений (голосователь) и 3 независимо разработанных версий работы. Все 3 версии работают параллельно (возможно, с некоторым временным сдвигом) на разных процессорах, результат их работы обрабатывает голосователь. Время работы голосователя определяется параметром <время_работы_голосователя>. Тот результат, который выдаёт б**о**льшая часть версий, принимается за финальный. Считать, что время работы версий одинаково, а данные от других работ поступают всем версиям работы мгновенно.

3.4 Восстановление блоками (RB_1_1)

Параметры:<время_тестирования>

Данный МОО включает модуль принятия решений (контрольный тест) и две версии работы. Когда первая из версий завершает свою работу, то результат её работы тестируется контрольным тестом. Если результат теста оказался неудачным, то запускается на выполнение следующая версия работы. Процесс продолжается пока результат одной из версий не будет принят, либо результат работы всех версий не будет отклонён. При построении расписания считать, что тест после функционирования первой версии работы всегда выдает результат “неуспешно”, а после функционирования второй версии работы всегда выдает результат “успешно”.

3.5 Контрольные точки (CP)

Параметры:<время_между_КТ>,<время_установки_КТ>

Данный МОО функционирует следующим образом. Через время определяемое параметром <время_между_КТ> работа прерывается и в течении времени определяемом параметром <время_установки_КТ> её состояние сохраняется на надёжном запоминающем устройстве (контрольная точка КТ). После сохранения КТ функционирование работы продолжается с момента прерывания. В конце выполнения работы также ставится КТ по той же схеме. В случае обнаружения ошибки в работе, она откатывается до ближайшей КТ. При построении расписания считать, что программа работает безошибочно и откатов не происходит.