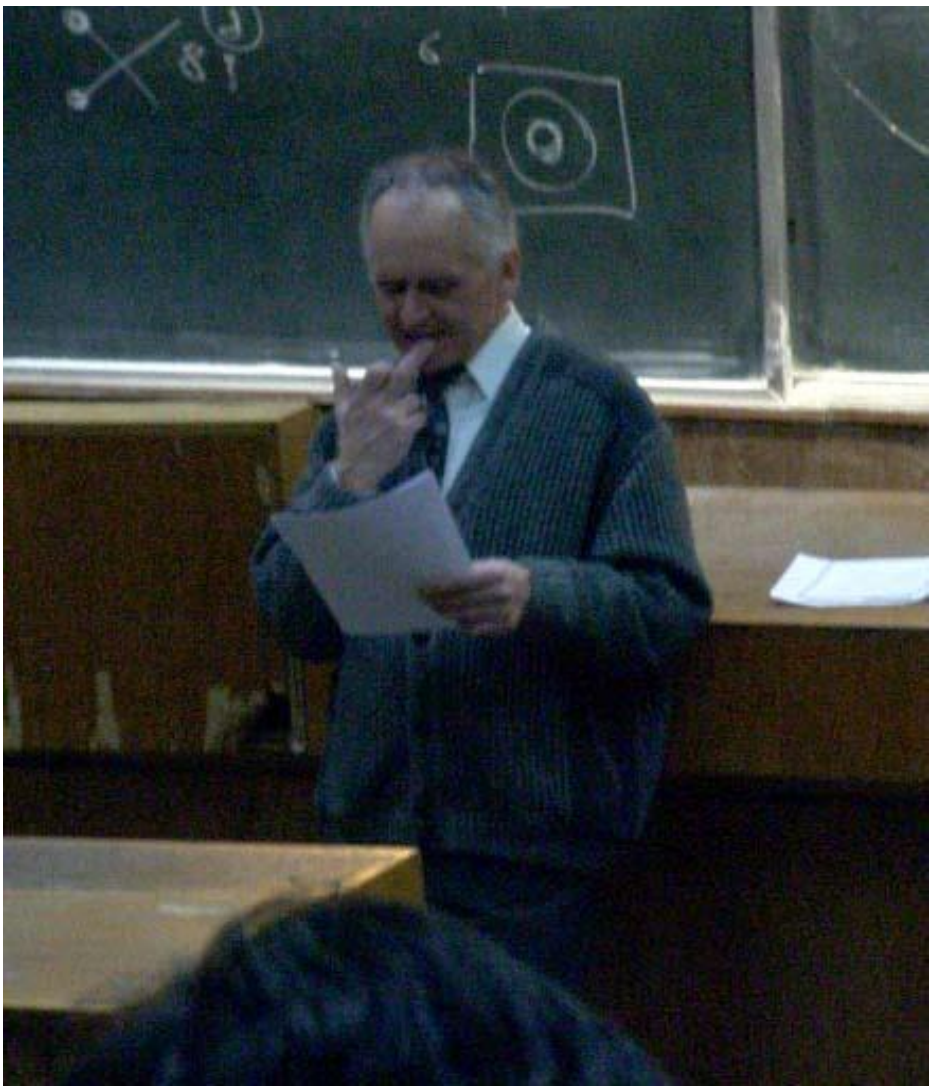
НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ УКРАИНЫ
«КИЕВСКИЙ ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ»

# *Раздуплятор по Симону*

**«AVE SIMONENKO»**

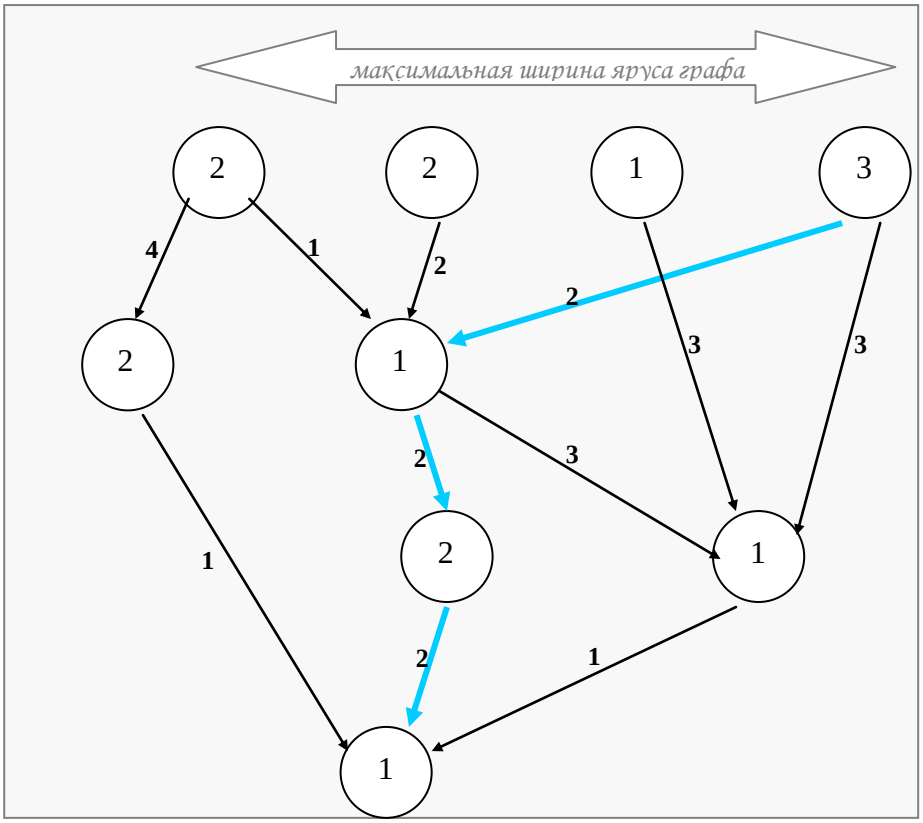
*v. 0 .9.5*





ИЗДАНО ПОД СОАВТОРСТВОМ MUROMTS'A

*I. В данной главе описан нереально-упрощенный пример задачи с экзамена по Симону. Задание звучит приблизительно следующим образом:* ***найти зоны оптимального поиска решения по заданному графу.***




*ЗЫ: критический путь – эт путь, по которому мы в суме получаем максимальные веса вершин (на графе отмечен жирными голубыми стрелками).*

*ЗЗЫ: весы вершин можно увидеть в кружочках графа, весы пересылок – над переходами.*

***Собственно решение данного трабла:***

$T_{кр}$(критическое)=7 (весы вершин критического пути: 3+1+2+1=7, на графе выделен голубым);

$T_{MAX}$ =15 (весы всех вершин графа: 2+2+1+3+2+1+2+1+1=15);

$N_{LOW} = | T_{max}/T_{кр} | = 15/7=2$;

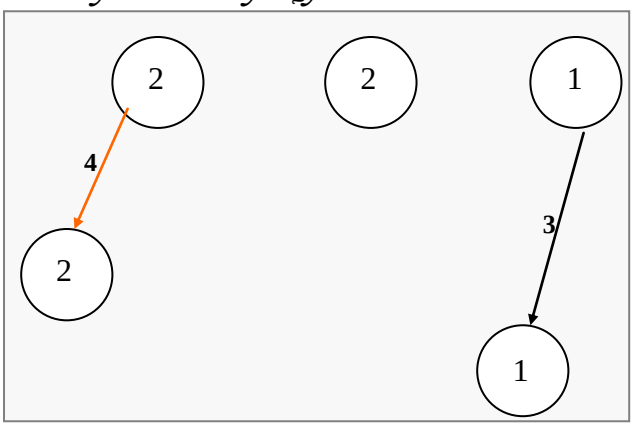
$N_{HIGH}$= 4 (максимальная ширина яруса графа);

$T_{MIN}$ = 6 (весы пересылок в критическом пути: 2+2+2=6);

$T_{MAX}$ = 24 (весы всех пересылок в графе);

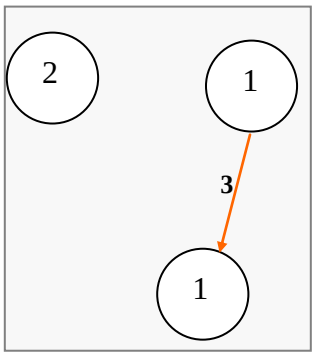
$N_{MAX}$ = 9 (количество всех вершин в графе).

*Теперя нужно сделать такую мутку под названием* ***зануление*** *всех переходов критического пути, енто делается очень просто – весы пересылок критического пути заменяем на нолики. Поскольку весы пересылок равны нулю, то и соответствующие связи идут нах. Получаем следующую бню:*




*ЗЫ: думаю Симону такая штуковина должна понравицца. ☺*

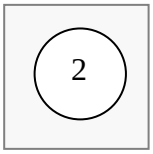
*Далее сново зануляем вышенацарапаную бень :*




*Для тех, кто в бронепоезде объясняю популярно, что здесь мы проводили последующее зануление оставшихся вершин, соответственно в итоге у нас получилась одна вершина, в которой нечего занулять.*

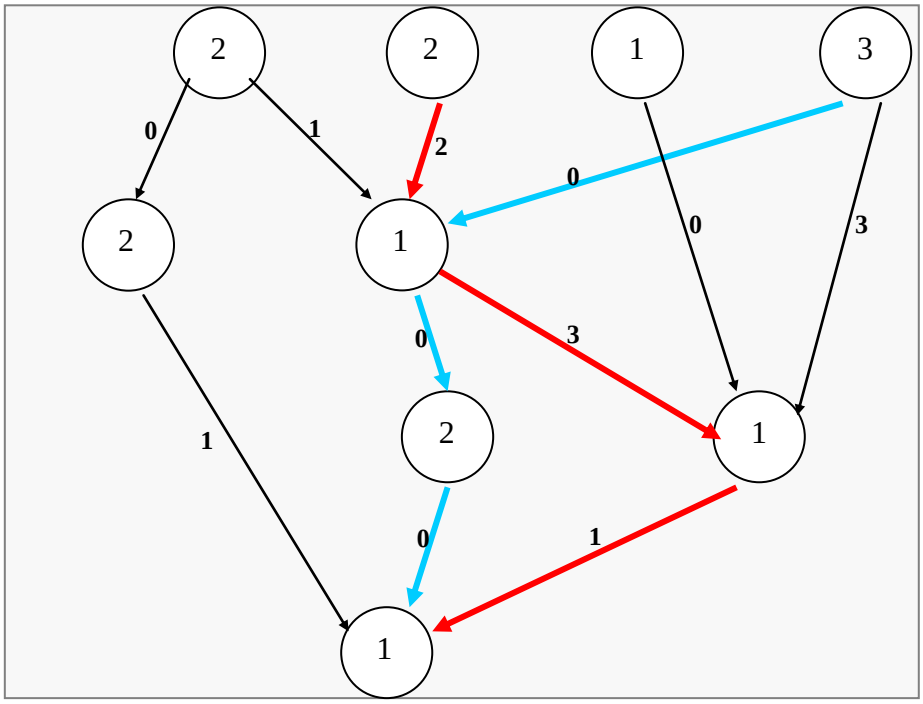
*ПЫСЫ: только не спрашивайте мну по какому принципу делаются эти зануления… но есть подозрения, что нам нуна найти минимальный критический путь…*

*Исново :*




*Вот такие вот пироги…*

*Для человекоф с тугоразвитой фантазией изобразим то, что у нас таки получилось :*

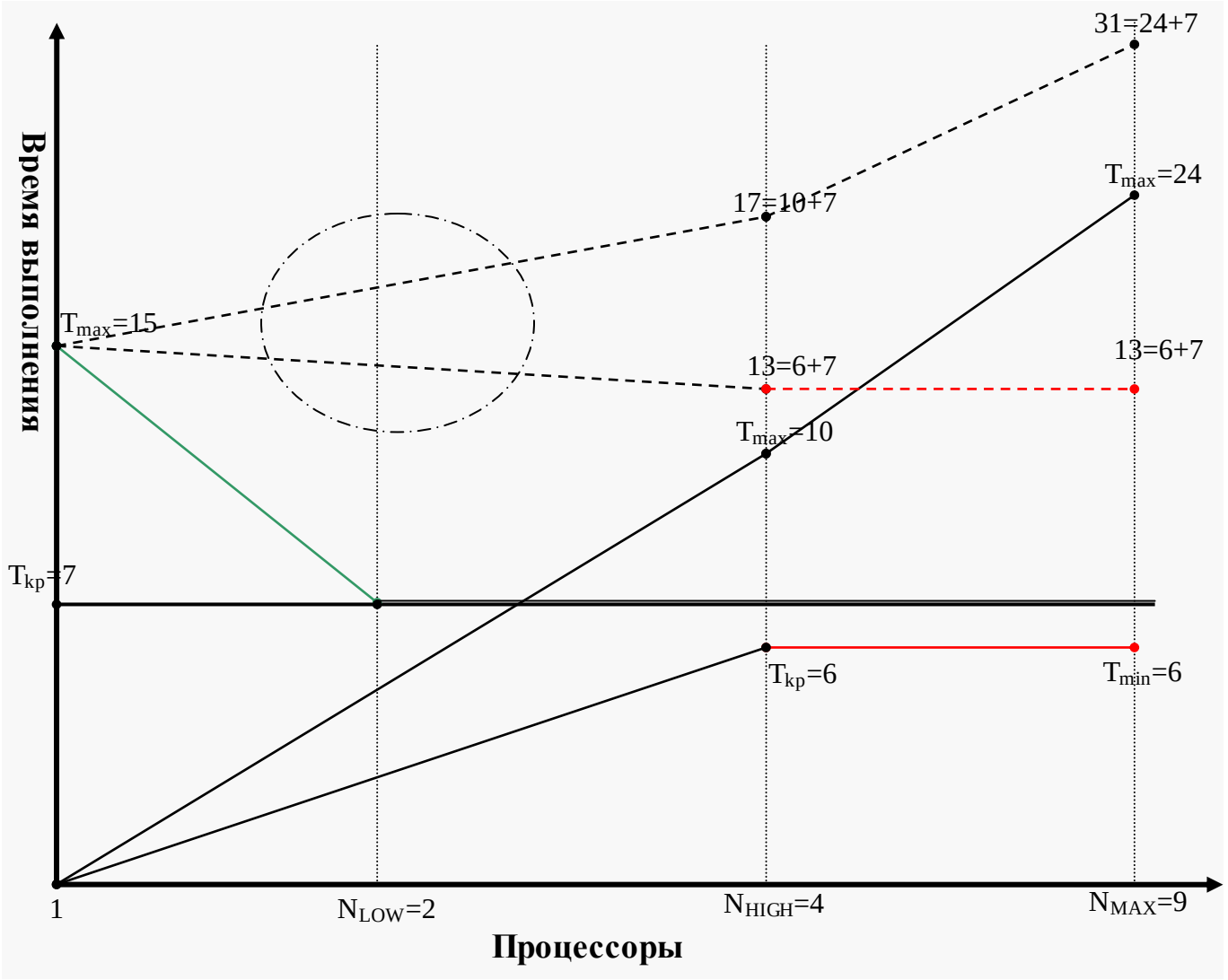



*Имеются подозрения, что этот критический путь, кстати, если вы еще не догадались – он выделен красным, стоит выбирать по максимальному весу пересылок. ИМХО…*

*Далее находим еще один критический путь (по пересылкам!!), на графе он указан красным цветом.*
*Продолжаем ~~страдать херне~~ решение :*

$T_{кр}$(критическое)=6 (сумма весов пересылок критического пути: 2+3+1=6 );
$T_{MAX}$ =10 (сумма весов всех пересылок графа: 1+2+3+3+1=10).

*Думаете фсе?? Не-а!! Теперь начнетса самое интересное – проэкт «малевалки»…*




-- *внимание ПАЛИВО!!*

*ПЫСЫ: пример приведен лишь в ознакомительных целях, все багги, глюки и совпадения являются случайными, и к автору никакого отношения не имеют.*

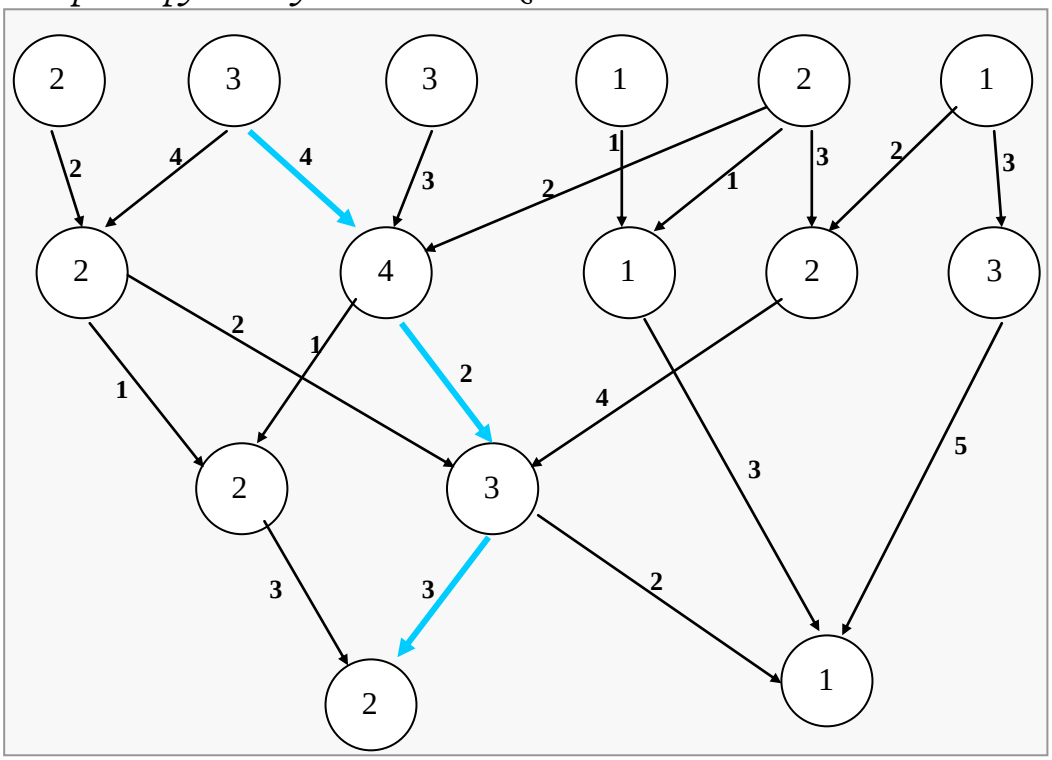
*ТЫ: если на экзамене, отрезки графика, отмеченные красным цветом, окажутся прямыми линиями, можете молча собрать вещи, выйти из лаборатории и готовиться к следующей передаче, которая назначена на 4/07 в 10:00, ИМХО* $T_{kp} = T_{min} = 6$ *полный бред!*

*ТЫЫ: при значении* $N_{LOW} = 3$ *на участке графика выделенного зеленым будет присутствовать изгиб, при этом координаты точки изгиба будут иметь следующий вид* $((N_{LOW}/2); (T_{max}/2))$*;*

*ТЫЫЫ: яйцеобразной фигурой на графике изображено решение – оптимальное количество процессоров, только как оно определяется и кому это надо для автора осталось загадкой…*

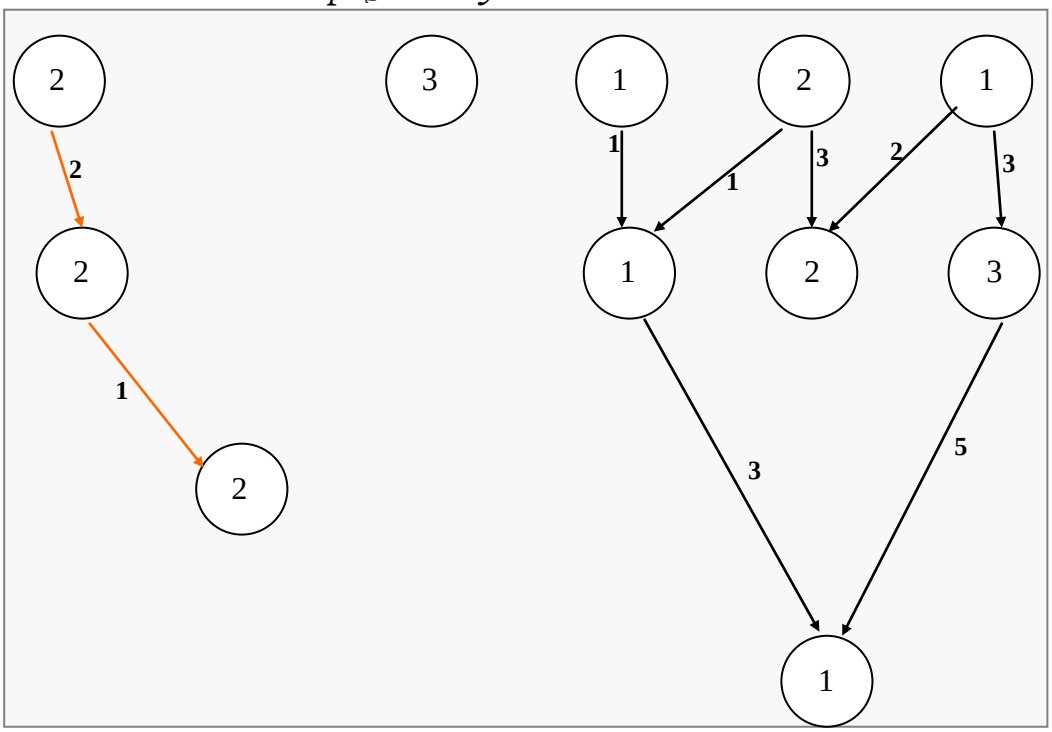
*ОТ АВТАРА: в реальной задаче на вы будите лицезреть граф как минимум на 40 вершин (а-ля фих просцыш…), поэтому, чтобы не получить сердечный приступ прямо на экзамене и скоропостижно не скончаться на своем графе, настоятельно рекомендуется прорешать около 50 графом с количеством вершин не менее 50, тщательно заполненных на рандомайзе человеком, который в этом абсолютно не шарит (мама, папа, сестра, кот…).*

*II. Решение следующего примера было любезно предоставлено Miromts'ем, который, однако, не гарантирует отсутствия ошибок☺*

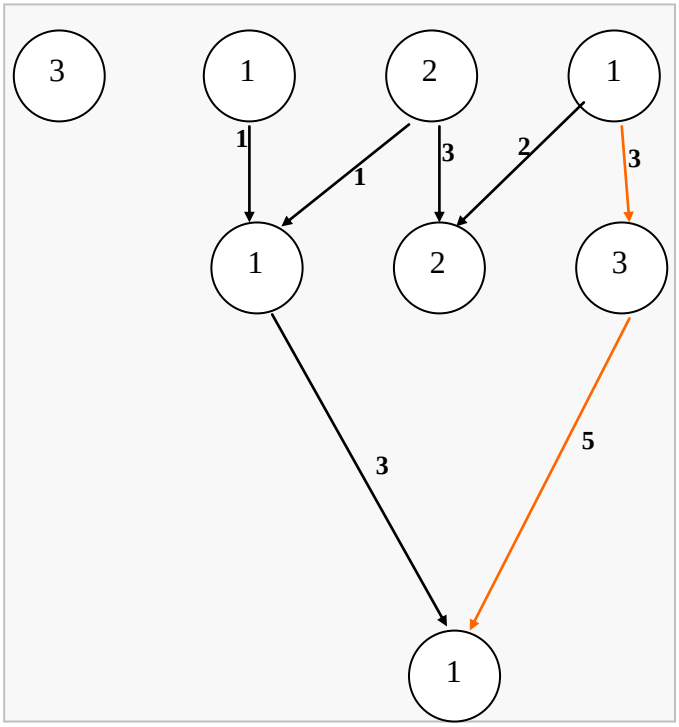


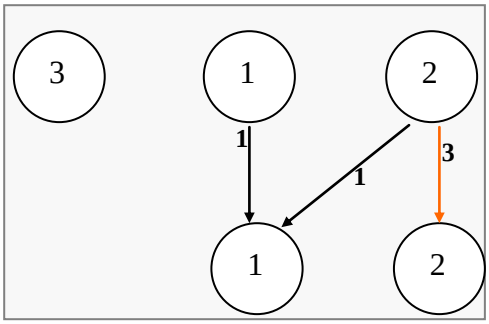

$T_{кр}$=12 (весы вершин критического пути: 3+4+3+2=12);
$T_{MAX}$ =32 (весы всех вершин графа);
$N_{LOW}$ = | $T_{max}/T_{кр}$ | =32/12=3;
$N_{HIGH}$= 6 (максимальная ширина яруса графа);
$T_{MIN}$ = 9 (весы пересылок в критическом пути: 4+2+3=9);
$T_{MAX}$ =51 (весы всех пересылок в графе);
$N_{MAX}$ =15 (количество всех вершин в графе).

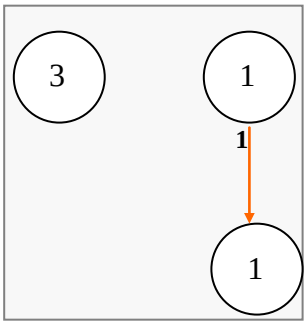
*Операция «зануление»:*




⇩



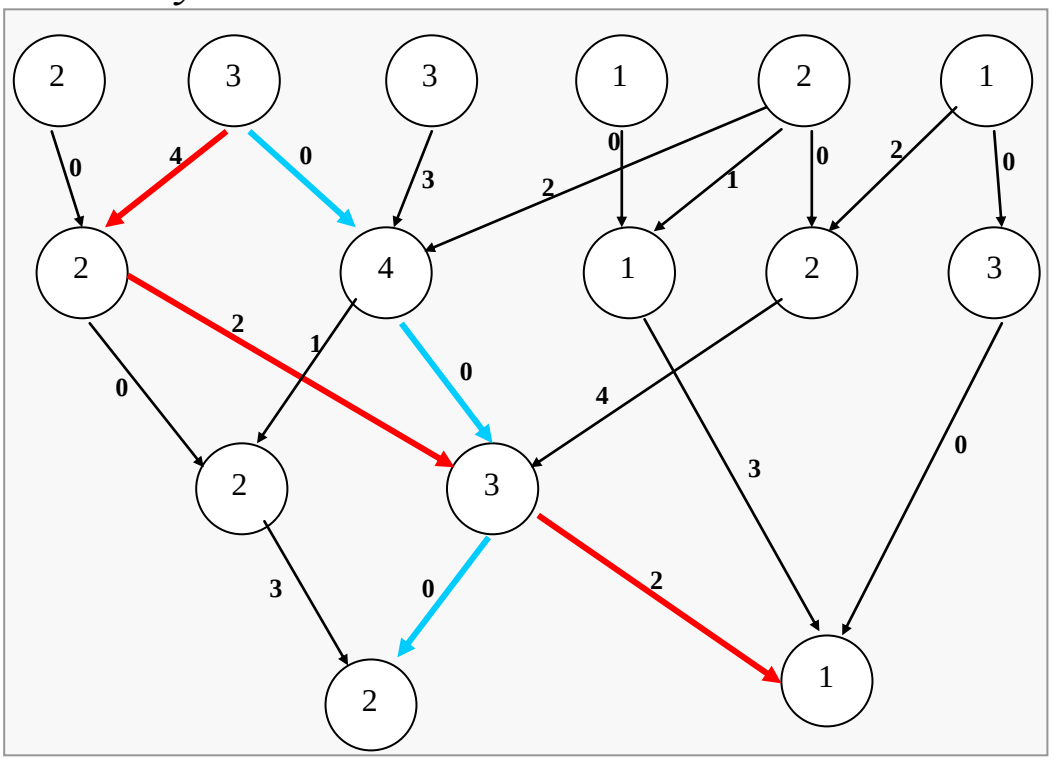


*<u>Алгоритм зануления, разработан Muromts'ем, дополнен Nicklays'ом</u>:*

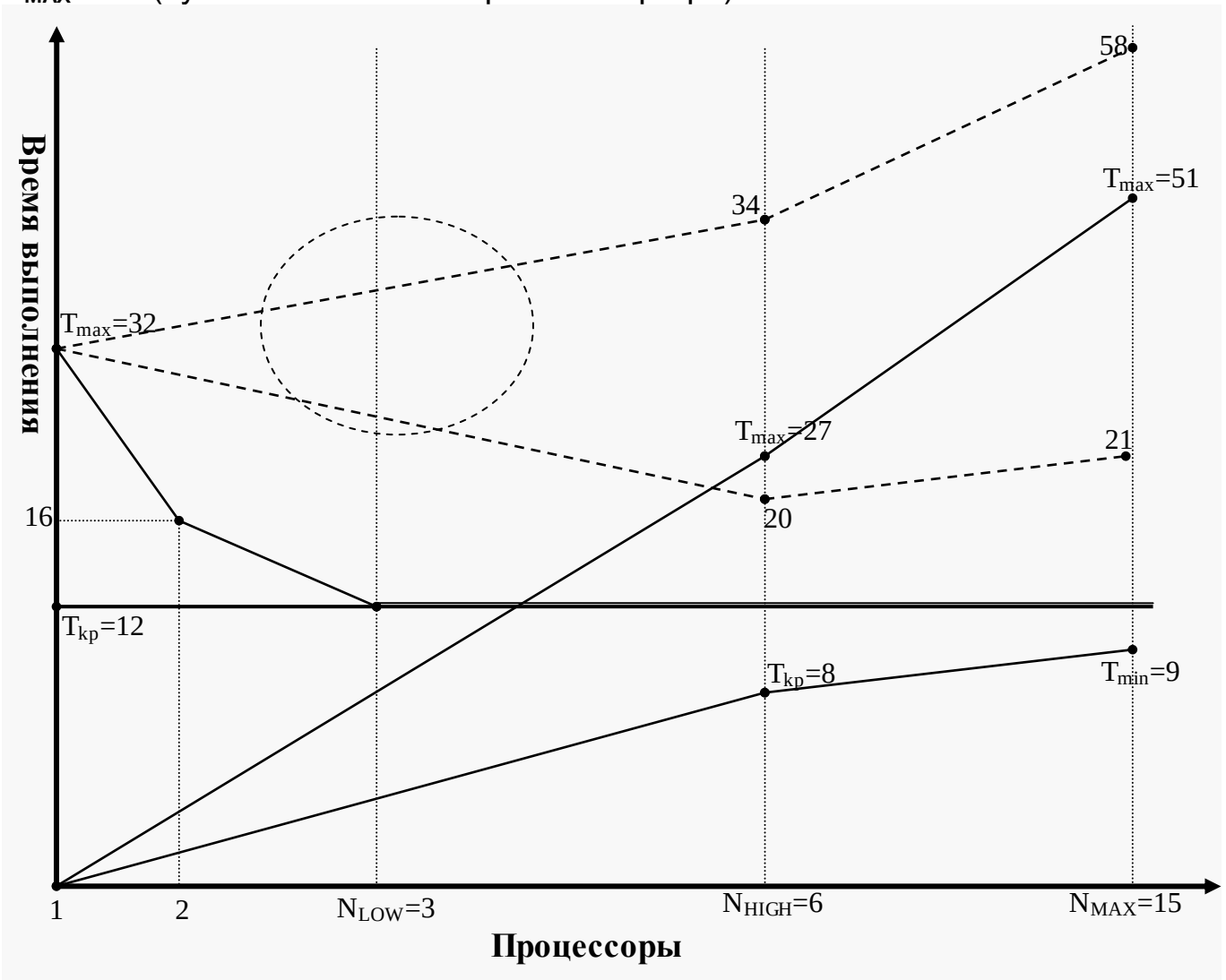
1. *Зануляем главный критический путь.*
2. *Ищем следующий критический путь, в оставшейся части графа и зануляем его.*
3. *Операции поиска и зануления продолжаем до тех пор, пока не будет занулен последний критический путь.*
4. *При наличии двух и больше равносильных критических путей, выходящих из одной вершины, выбор делаем произвольно.*

*По неподтвержденным данным оставшаяся вершина графа указывает на оптимальное количество процессоров, которые будут задействованы при решении…*

*В итоге получаем вот что:*




$T_{кр}$=8 (сумма весов пересылок критического пути: 4+2+2=8 );
$T_{MAX}$ = 27 (сумма весов всех пересылок графа).



*III. В этой главе предоставлены ответы на вопросы с билетов, которые удалось достать с экзамено по СПО в 2005 году.*

==========Билет №6==========

***1. Классификация ОС.***

*ОС -- совокупность программ, обеспеч. эффективную работу оборудования, решающих любую задачу.*

*-однопрограммные: система работает в однопрограммном режиме, все ресурсы системы отданы одной задаче, которая не может быть прервана и завершается аварийно или по окончании;*

*-многопрограммные: система работает в многопрограммном режиме, если есть несколько задач на разной стадии выполнения, каждая задача может прервана и востановлена;*

*-однопроцессорные: организация вычислит. процесса с планированием во времени;*

*-сетевые;*

*-многопроцессорные: планирование во времени и пространстве;*

*-распределенная: обеспеч. работу клиента в сети, поддержка иллюзии работы на 1 машине;*

*-виртуальные: позволяют работать терминалам в той ОС, в которой захочет юзер;*

*-офисные;*

*-кластерные.*

***2. Доказать теорему о "Конфликтных назначениях."***

*Если в матрице МТ[i,j], i=1..N, j=1..N, можно выделить несколько подматриц, удовлетворяющих Теореме 5, то все соответствующие симметричные им, относительно главной диагонали, подматрицы являются "конфликтными" и должны быть обнулены.*

*Доказательство.*

*Предложим, что в RJ имеется подматрица МТ, в которой ∃МТ[i*,j*]=1 и (i*,j*)∈A, где i*∈{(T+1),..,N}, j*∈{1,..,(N-S)}. Выделение подматрицы МТ делит МС на подматрицы: S×T, (N-S)×T, (N-T)×S, (N-T)×S. Так как по условию S+T=N, то подматрицы (N-S)×T и (N-T)×S квадратные. Тогда предположение, что ∃(i*,j*)∈A, где i*∈{(T+1),..,N}, j*∈{1,..,(N-S)} приводит к тому, что в подматрице (N-S)×N должны присутствовать (T+1) назначений, входящих в A. Но так как (T+1)>(N-S) и S+T=N, то это предположение не верно. Аналогичное доказательство можно привести и для подматрицы (N-T)×S. Теорема доказана.*

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 0 | 1 | 0 |
| 2 | 0 | 1 | 1 | 1 | 1 | 1 |
| 3 | 1 | 0 | 1 | 1 | 1 | 0 |
| 4 | 1 | 1 | 1 | 1 | 0 | 1 |
| 5 | 1 | 0 | 1 | 0 | 1 | 0 |
| 6 | 1 | 0 | 1 | 0 | 1 | 0 |
| | 1 | 2 | 3 | 4 | 5 | 6 |

Рис. 2.22.а Исходная матрица связности

| | | | | | | |
|---|---|---|---|---|---|---|
| 6 | 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 5 | 1 | 1 | 1 | 0 | 0 | 0 |
| 3 | 1 | 1 | 1 | 1 | 0 | 0 |
| 2 | 1 | 0 | 1 | 1 | 1 | 1 |
| 4 | 1 | 1 | 0 | 1 | 1 | 1 |
| | 3 | 1 | 5 | 4 | 2 | 6 |

Рис. 2.22.б Матрица после преобразования

| | | | | | |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 0 | 1 | 1 |

Рис. 2.22.в Матрица после коррекции

==========Билет №7==========

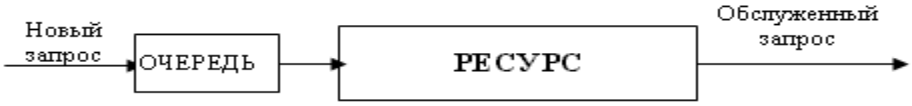
***1. Дисциплины обслуживания заявок - их виды и особенности.***

*а) Безприоритетные: принцып обслуживания заявок – по приоритету.*

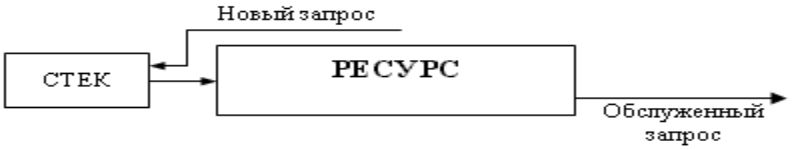
*-- случайная выборка – заявки обслуживаются по случайной выборке, на рандомайзе;*

*-- FIFO (First In - First Out) - первый пришел - первый обслужен. Схема доступа - очередь.*

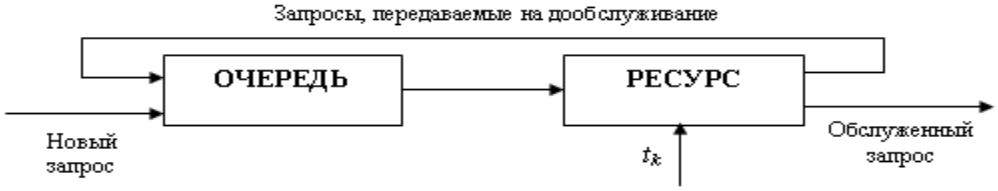
*Время нахождения в очереди длинных (то есть требующих большого времени обслуживания) и коротких запросов зависит только от момента их поступления.*



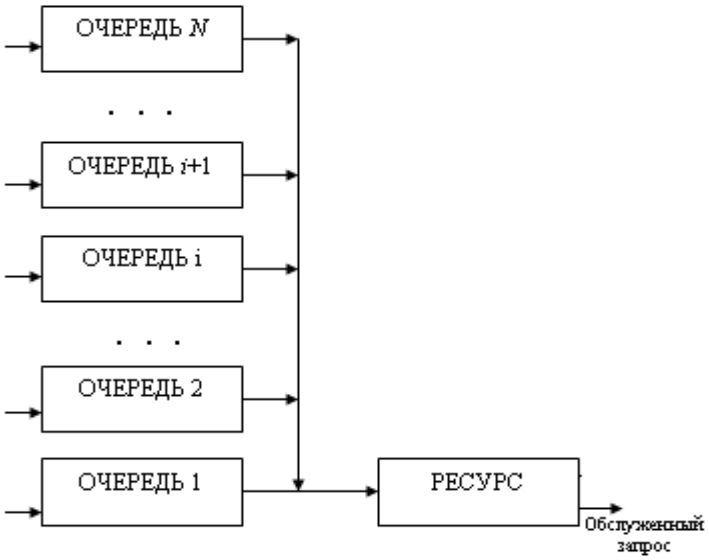

*-- LIFO (Last In - First Out) - последний пришел - первый обслужен. Схема доступа - стек.*



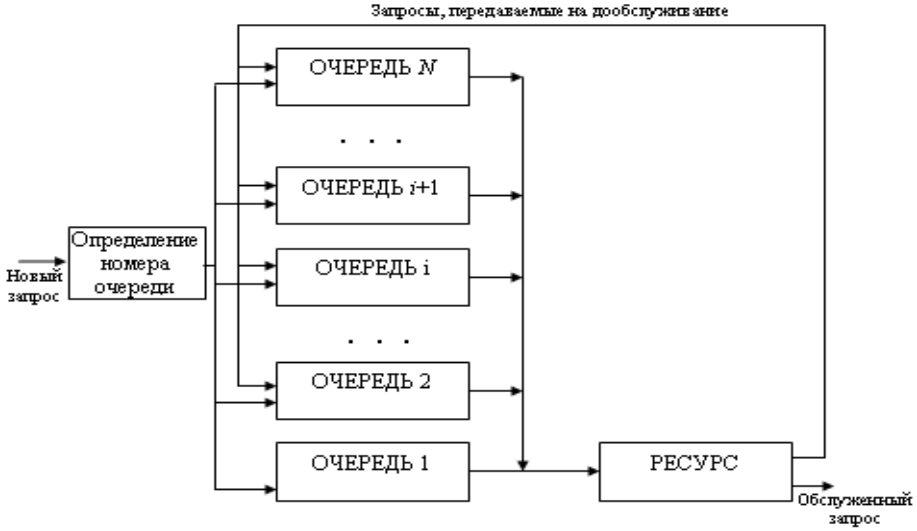

*-- RR (Robin Round) круговой циклический алгоритм. Запрос обслуживается в течение кванта времени tk. Если за это время обслуживание не завершено, то запрос передается в конец входной очереди на дообслуживание.*
*Здесь короткие запросы находятся в очереди меньшее время, чем длинные.*




*-- FBN многоприоритетный алгоритм обслуживания с несколькими очередями, все заявки поступают в первую очередь, если заявка большая – она идет во вторую очередь, если 1-я пустая, то со 2-й в 1-ю очередь. Маленькие заявки быстрей входят в систему.*




*-- Корбато программа сразу поступает в очередь i = [log2 lp/ltk + 1], где lp - длина программы в байтах; ltk - число байт, которые могут быть переданы между ОЗУ и внешней памятью за время tk. Эта дисциплина позволяет сократить количество системных переключений за счет того, что программам, требующим большего времени решения, будут предоставляться достаточно большие кванты времени уже*



*при первом занятии ими ресурса (нерационально программе, которая требует для своего решения 1 час времени, первоначально выделять квант в 1 мс).*
*б) Приоритетные: обработка с относительным и абсолютным приоритетом.*
*-- абсолютный приоритет производится с вытеснением(заявка с высоким приоритетом вытесняет заявку с низким из ресурса);*
*-- относительный приоритет производится без вытеснения;*
*-- динамический приоритет смена по времени работы, смена по времени ожидания;*
*-- смешанные использование как относительного, так и абсолютного приоритета.*

***2. Венгерский метод.***

*Идея -- поиск минимальной суммы элементов. Временная сложность 0 [$n^3$]*
*Задание планирования сводится к поиску максимального паросочетания во звешенном двудольном графе, надо найти максимальные паросочетания, чтобы сумма назначения была максимальна или минимальна.*

1) *В каждом столбце матрицы находим минимальный элемент и отнимаем от каждого элемента этого столбца.*
2) *В каждой строке матрицы находим минимальный элемент и отнимаем от каждого элемента столбца.*
3) *В каждой строке и столбце должен оказаться как минимум один «0». Из этой матрицы забираем все «0» и заменяем на «1».*
4) *Находим максимальные паросочетания (если мы получим совершенный вариант, то переносим его на первую матрицу и получим минимальную сумму).*
5) *На этой матрице отмечаем нули, которые вошли в решения и те, которые не вошли. Отмечаем строки, в которых нет отмеченных «0», а есть зачеркнутые.*
6) *Отмечаем те столбцы, в которых есть зачеркнутые нули, отмеченных строк.*
7) *Отмечаем строку, содержащую отмеченный «0», который содержит строку, отмеченную в придыдущем шаге.*
8) *Отмечаем столбец, в котором есть зачеркнутый «0», который содержит строку, отмеченную в придыдущем шаге.*
9) *Отмечаем пунктиром помеченные столбцы и непомеченные строки.*
10) *Выписываем элементы, через которые проходят пунктирные линии и среди них ищем минимальный элемент.*
11) *Отнимаем этот минимальный элемент от тех столбцов матрицы через которые не проходят пунктирные линии.*
12) *Прибавляем этот элемент к тем строкам, через которые проходят пунктирные линии.*
13) *Объеденные единицы переносим на исходную матрицу.*

*Далее приведено более научное объяснение Венгерского метода а-ля по Симону:*

- *(1) Формирование "ящика": необходимо представить данные о M заданиях и N ресурсах в виде матрице размером M*N. Назовем ее матрицей отношения заданий-ресурсов RJ[M,N].*
- *Определение масштаба оптимизации:*
  - *n=max(M,N) если M<N;*
  - *n=min(M,N) если M>N;*

  *Масштаб оптимизации позволяет привести в соответствие численные отношения ЗАДАНИЯ-РЕСУРСЫ и исходные данные к квадратной матрице n*n. Каждый элемент матрицы соответствует весу назначения i-j, i=1..n, j=1..n.*

*После этого переопределяются значения элементов матрицы RJ[i,j]= μ-RJ[i,j], i=1..n, j=1..n, где μ есть некоторое заданное число (достаточно большое чтобы RJ[i,j]>0). Мы получаем новую квадратную матрицу RJ[n,n]. Операция переопределения выполняется для исключения из рассмотрения назначений, выполнение которых в реальной системе принципиально возможно, и требует больших затрат.*

- *(2) Определение "внешней границы" зоны поиска: в Венгерском методе, роль "внешней границы" зоны поиска играют*

*минимальные элементы матрицы RJ, которые определяются следующим образом:*

- *Для каждой строки и каждого столбца матрицы RJ, определяются элементы с минимальным весом.*
- *Определенные по пункту 1 элементы вычитаются из всех элементов данной строки или столбца (операция выполняется сначала для всех строк, а затем столбцов).*

*Эти минимальные элементы принимаются как элементы "внешней границы" зоны поиска. Зона поиска определяется нулями матрицы RJ обозначается как "текущая линия поиска (ТЛП)" .*

■ *(3) Поиск набора "назначения-шариков" на текущей линии поиска: для полученного множества минимальных нулевых элементов на ТЛП, а это множество и есть возможный вариант решения, производится поиск максимального паросочетания для двудольного графа, отображающего нулевые элементы матрицы RJ. Для нахождения максимального паросочетания используется адаптивный алгоритм мультианализа (АМА), теоретическая основа которого представлена в главе 2 (где роль "1" играют "0") и изложена в работах автора [34,38,39].*

*Если удается найти совершенное паросочетание размером n, то поиск прекращается и переходим на пункт (5). В противном случае переходим к пункту 4.*

■ *(4) Определение новой линии поиска: если на текущей линии поиска не существует совершенного паросочетания n, то определяется новая линия поиска (НЛП) следующим образом:*

- *Для найденного максимального паросочетания (которое не является полным) его определяется "минимальная опора".*
- *На основе "минимальной опоры" производится перестановка нулей и определяется новая совокупность "0" в матрице RJ. Эта совокупность и есть НЛП для нового поиска наборов "шариков-назначений" (полного максимального паросочетания).*

*После определения НЛП, переходим к процедуре (3) и повторяем все действия, описанные выше, до тех пор, пока не найдется набор, имеющий совершенное паросочетание.*

*(5) Вывод оптимального расписания: после определения набора назначений (совершенного паросочетания для двудольного графа состоящего из "0" на ТЛП), он выводится как расписание оптимального распределения заданий и ресурсов в данной НСРОД.*

==========Билет №9==========

## 1. Приоритетные дисциплины обслуживания.

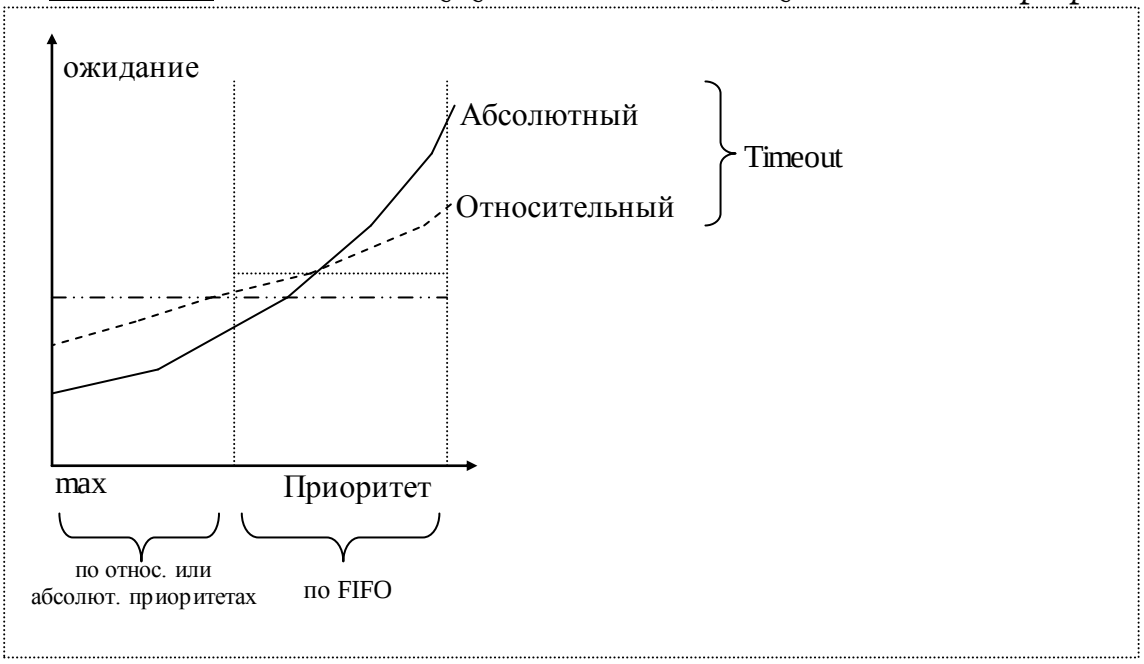
*Приоритетные дисциплины -- обработка с относительным и абсолютным приоритетом.*

*-- абсолютный приоритет производится с вытеснением(заявка с высоким приоритетом вытесняет заявку с низким из ресурса);*

*-- относительный приоритет производится без вытеснения;*

*-- динамический приоритет смена по времени работы, смена по времени ожидания;*

*-- смешанные использование как относительного, так и абсолютного приоритета.*




## 2. Оценка методов поиска максимального паросочетания.

*Анализ алгоритмов поиска максимального паросочетания, а также анализ процесса поиска решения в выделенных зонах показывает, что наибольшие трудности, влияющие на количество шагов, а отсюда и на время поиска максимального паросочетания, возникают в двудольных невзвешенных графах. Эти трудности вызваны тем, что поиск*

*максимального паросочетания основан на центральной теореме Кенига-Холла о существовании паросочетания и тереме Бержа. По* ***теоремой Бержа — "паросочетание*** *M в графе G максимально тогда и только тогда, когда в G не существует увеличивающего пути относительно M". Поэтому, все известные алгоритмы предусматривают выполнение попыток поиска увеличивающего пути от свободных вершин после генерирования базового варианта даже в том случае, если этого пути нет, что существенно увеличивает время поиска. Кроме этого, заранее неизвестно чему равна мощность паросочетания, а известные алгоритмы вычисления мощности максимального паросочетания по временной сложности превышают поиск самого максимального паросочетания.*

==========*Билет №13* ==========

***1. Понятие задания, задача, программа, данные объекты ОС.***

*Задание – внешняя единица работы системы, на для работы которой система ресурсы не выделяет.*

*Задача – внутренняя единица работы системы, для которой система выделяет ресурсы. Задача должна быть зарегистрирована в системе и для нее должны быть созданы системные объекты.*

*Процесс – траектория процессора в адресном пространстве. Смена состояния блока управления задачей называется процессом.*

*Программа – это ресурс ОС, который подчиняется определенной задаче.*

*Данные – это ресурс ОС, который подчиняется только программе.*

***2. Доказать теорему о вычислении мощности паросочетаний.***

*Если в матрице связности (МС), отображающей двудольный граф размерности N можно выделить нулевую подматрицу S×T, где S+T>N, то задача не имеет варианта полного решения и мощность паросочетания (M) равна 2N-(S+T).*

| | N-S | | | S | | | | |
|---|---|---|---|---|---|---|---|---|
| | 1 | 0 | 1 | 0 | 0 | 0 | 0 | |
| | 0 | 1 | 1 | 0 | 0 | 0 | 0 | T |
| T | 1 | 1 | 1 | 0 | 0 | 0 | 0 | |
| | 1 | 0 | 1 | 0 | 0 | 0 | 0 | |
| | 1 | 1 | 0 | 0 | 0 | 0 | 0 | |
| N-T | 1 | 0 | 1 | 1 | 1 | 0 | 1 | N-T |
| | 0 | 1 | 1 | 1 | 0 | 1 | 1 | |
| | N-S | | | | S | | | |

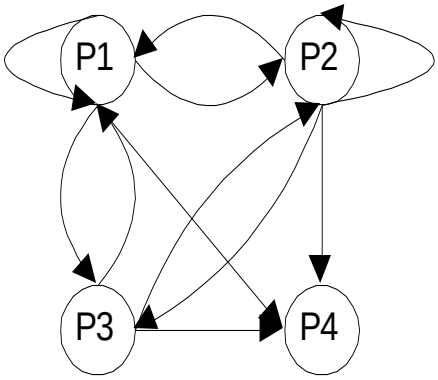
*Доказательство.*

*Пусть A — неотрицательная матрица порядка N. Тогда per(A)=0 в том и только том случае, когда A содержит нулевую подматрицу размера S×T, где S+T=N+1. В том случае, когда per(A)=0, можно утверждать, что при данной исходной информации нет совершенного паросочетания, т.е. мощность паросочетания меньше N. В этом случае мощность паросочетания определяется количеством нулей главной диагонали, попавших в нулевую подматрицу S×T. Выделение нулевой подматрицы (S+T)>N разбивает МС на четыре подматрицы S×T, (N-S)×T, (N-T)×S, (N-T)×(N-S) Используя термины заявки и ресурсы, можно утверждать, что т.к. подматрицы (N-S)×T, (N-T)×S не квадратные (по условию (S+T)>N), то T заявок не могут претендовать на захват (N-S) ресурсов, при T>(N-S), или T заявок не могут быть обслужены (N-S) ресурсами если T<(N-S). Аналогично, S ресурсов не могут обслужить (N-T) заявок при S<(N-T) или (N-T) заявок не могут быть обслужены S ресурсами если S>(N-T). Тогда можно сделать вывод, что T-(N-S) заявок не будут обслужены*

*и S-(N-T) ресурсов не будут заняты. Поскольку |T-(N-S)| =|S-(N-T)|, то мощность максимального паросочетания равна M=N-(S+T-N)=2N-(S+T).*

==========Билет №14 ==========

1. ***Состояния процессора условного перехода.***




*P1: выполнение задач ОС.*
*P2: выполнение задач счетчика.*
*P3: дешифрация прерываний.*
*P4: прерывания схем контроля.*
*P3 – если в системе фиксируются прерывания – система безусловно переходит в состояние P3, где и срабатывает. Системе важен приоритет: системный – P1, проблемный – P2. Сброс выполняется по кванту и выбирается процесс для обработки.*