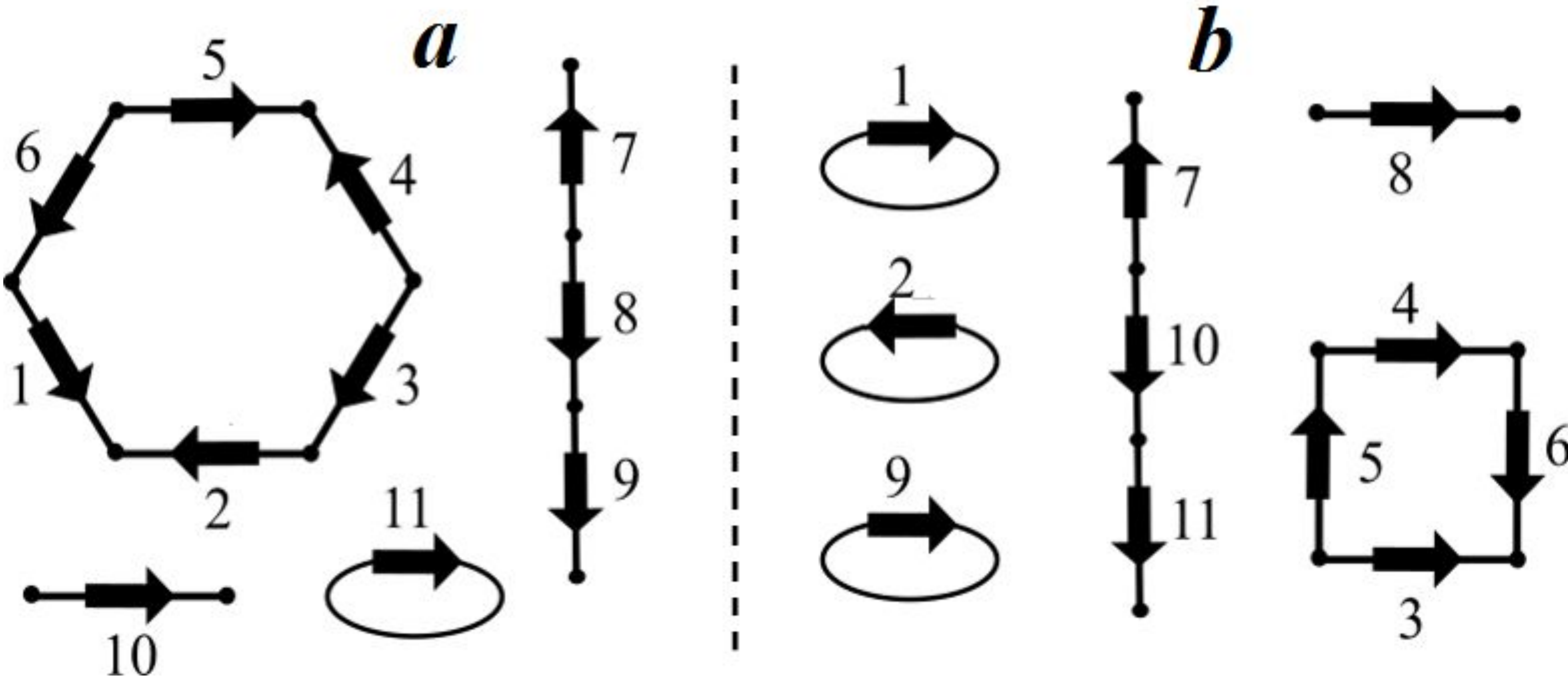
Определение. **Структура *a*** (или ***b***, ***c***, …) – ориентированный граф, состоящий из циклов (включая петли) и цепей длины ≥1 с именами рёбер (=натуральными числами). Это – нагруженный ориентированный граф, у которого все вершины степени 1 или 2.

ЗАДАЧА ПРЕОБРАЗОВАНИЯ для DCJ. Фиксированы 6 операций, см. их ниже. Даны переменные цены (>0) этих операций и переменные структуры ***a*** и ***b***. Линейным алгоритмом найти последовательность от ***a*** к ***b***, на которой достигается минимум её суммарной цены (**по всем последовательностям от *a*** к ***b***). Такую последовательность, как и её цену, назовём **кратчайшей**.

Пример таких структур *a* и *b*. Нужно найти кратчайшее преобразование *a* в *b* **указанными ниже операциями DCJ**:

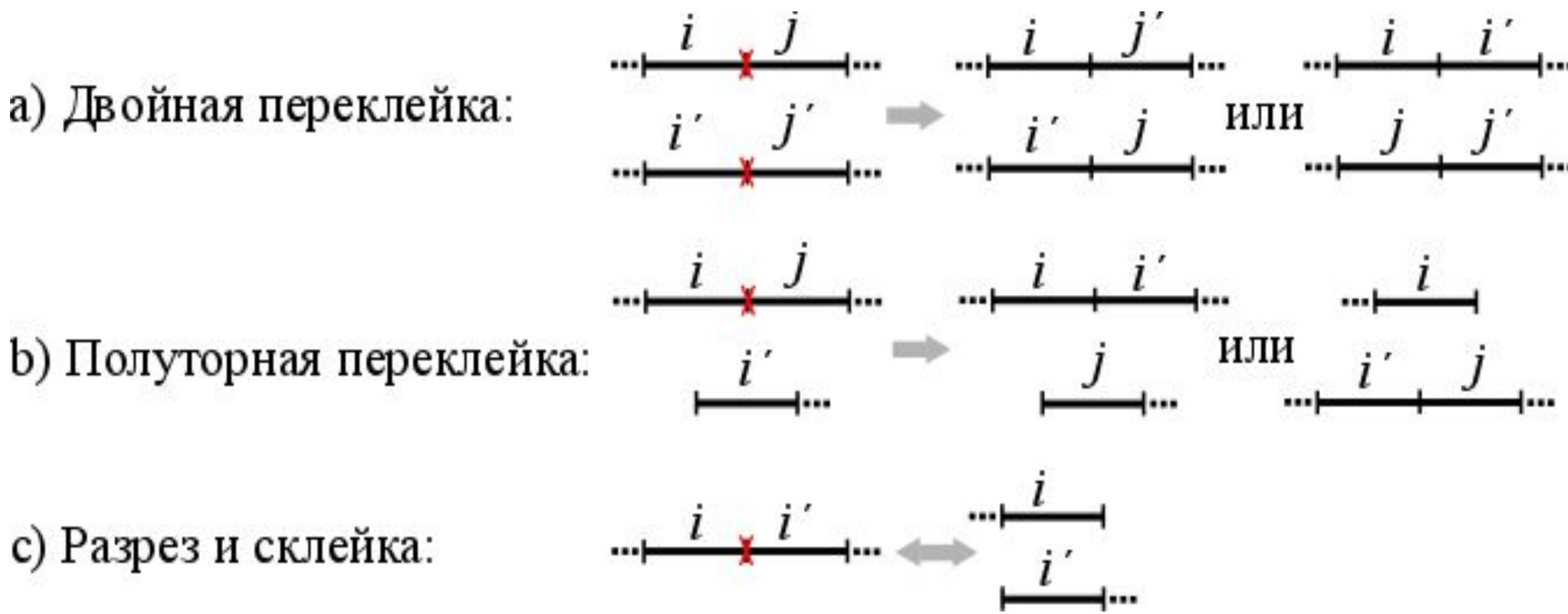


Набор операций **DCJ** для Задачи преобразования:

расклеить две вершины и склеить четыре образовавшиеся свободных (т.е. степени 1) края рёбер по-другому (двойная переклейка);

расклеить одну вершину и склеить один из образовавшихся краёв ребра с каким-то свободным краем (полуторная переклейка);

расклеить вершину или, обратная операция, **склеить** два свободных края (одинарные переклейки).



4 первых операции в задаче преобразования для DCJ :

расклеить какую-либо вершину (***Cut***),

склеить два свободных (т.е. степени 1) края (***OM***);

расклеить вершину (не на краю цепи) **и склеить** один из образовавшихся свободных краёв с уже свободным краем (***SM***);

расклеить две вершины (обе не на краю цепи) **и склеить** образовавшиеся свободные края (***DM***).

(***SM*** и ***DM*** – композиции ***Cut*** и ***OM***. В этом смысле хватило бы только операций ***Cut*** и ***OM***)

Эти четыре операции названы ***DCJ-операциями***, см. figure 1 в нашей статье в **arXiv**.

Разрешены ещё две операции для DCJ : **удалить** (***Rem***) связный участок рёбер с именами из *a*, но не из *b*; **вставить** (***Ins***) такой участок с именами не из *a*, но из *b*.

При удалении участка образовавшиеся свободные края склеиваются, а при вставке участка сначала вершина расклеивается (если она не крайняя) и после вставки две пары образовавшихся свободных краёв склеиваются.

Паралоги – рёбра в структуре с одинаковыми именами.

Состав структуры – множество имён в ней.

Теперь будут только две структуры ***a*** и ***b***.

до конца семестра будем изучать:

Точный линейный по времени работы <u>алгоритм</u> кратчайшего преобразования одной <u>структуры</u> в другую:

<u>**нер**авный</u> состав у ***a*** и ***b***, <u>**любы**е</u> цены операций,

<u>**пар**алогов</u> в ***a*** и ***b*** нет.

Здесь мы приведём строгое доказательство точности и линейности алгоритма из K.Yu. Gorbunov, V.A. Lyubetsky. Linear time additively exact algorithm for transformation of chain-cycle graphs for arbitrary costs of deletions and insertions. Mathematics, 2020, 8, 11.

Случай паралогов рассматривается в другой нашей статье:

Multiplicatively exact algorithms for transformation and reconstruction of directed path-cycle graphs with repeated edges. Mathematics, 2021, 9, No. 20, Art. 2576.

Русские предварительные вариант этих статей также лежит здесь.

Компьютерная программы для этих алгоритмов и для

алгоритма реконструкции очень востребованы!

Попробуйте ваши силы.

Для циклических структур Задача преобразования строго и

полностью рассмотрена в Лекции 9

– рекомендуем сначала изучить эту лекцию, и

только затем начинать материал 2-го семестра!

См. также статью аспиранта В. Новикова.

Ещё ссылки по этой задаче преобразования:

Горбунов К.Ю., Любецкий В.А. Почти точный линейный алгоритм преобразования графов из цепей и циклов, с оптимизацией суммы цен операций // Доклады Академии наук, 2020, том 494, № 6, стр. 26–29. (только формулировки)

K.Yu. Gorbunov, V.A. Lyubetsky. Linear time additively exact algorithm for transformation of chain-cycle graphs for arbitrary costs of deletions and insertions. Mathematics, Nov 10 2020, Vol. 8, No. 11, Art. 2001, 30 pp. (сложное доказательство)

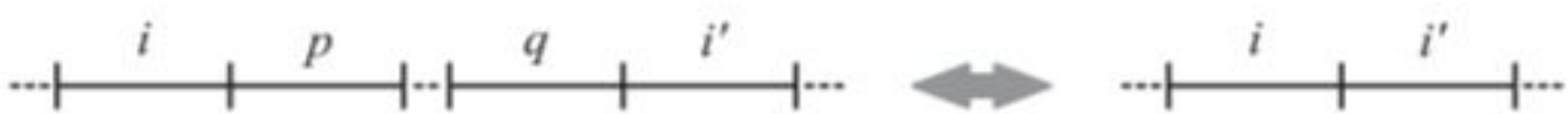
K.Yu. Gorbunov, V.A. Lyubetsky An almost exact linear complexity algorithm of the shortest transformation of chain-cycle graphs. Eprint, **arXiv**:2004.14351, Apr 29 2020. (здесь много полезных рисунков)

K.Yu. Gorbunov, V.A. Lyubetsky (2017). Linear algorithm of the minimal reconstruction of structures. Probl of Inform Transmission 53(1):55–72. (особо просто и на русском)

Примеры этих операций над структурой ***a*, которые** последовательно преобразуют ***a*** в структуру ***b***:

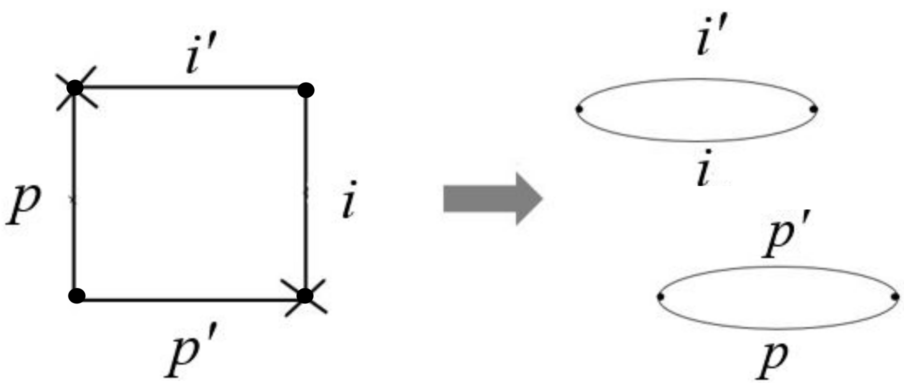
1) **удалить** связный участок рёбер с именами из ***a***, но не из ***b***,

2) **вставить** связный участок рёбер с именами из ***b***, но не из ***a*** :

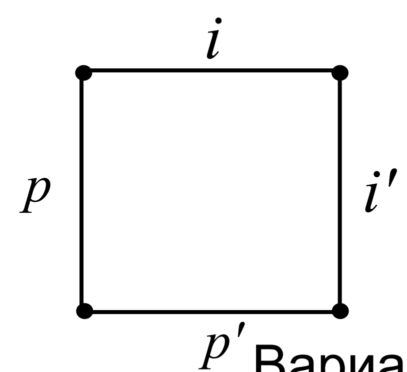


3) **переклейка**: расклеить две вершины и склеить по-другому образовавшиеся свободные вершины

(= края степени 1): пусть разрезы в **одном** цикле

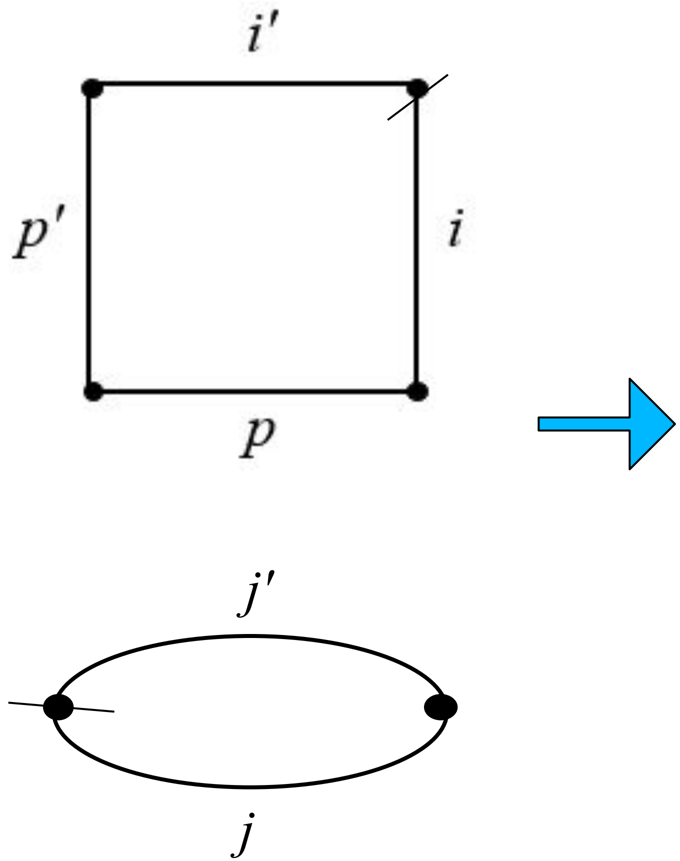


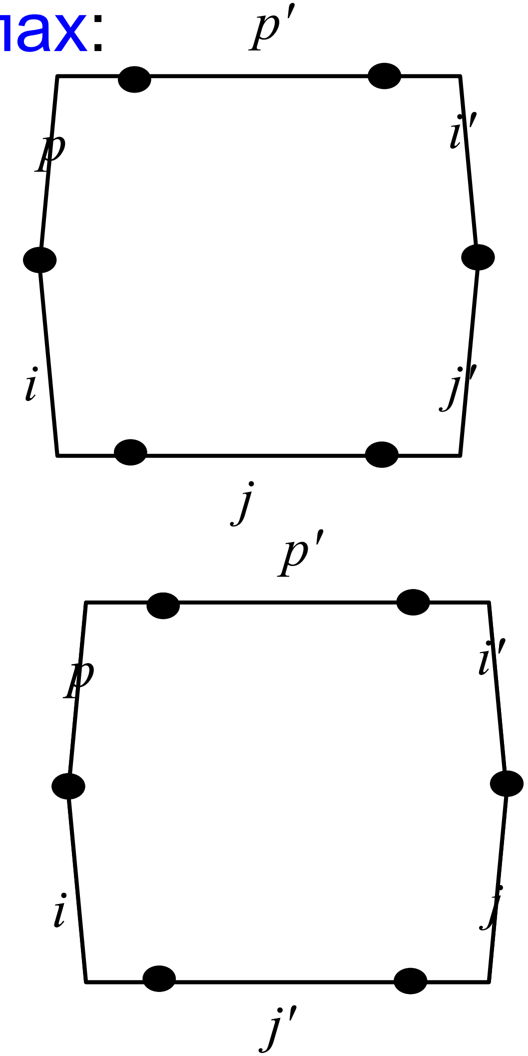
Вариант 1



Вариант 2

Пусть разрезы в разных циклах:





- вариант 1

- вариант 2

Каждой операции приписана **цена** – строго положительное рациональное число. В Лекции 9 алгоритм приведён для равных цен, т.е. каждая операция имеет цену равную 1.

Начало ребра с именем **5** помечаем **5_1** , а его конец помечаем **5_2**.

<u>**Важное определение!**</u> Общий граф ***G*** <u>**получается из пары**</u> **структур** <***a,b***> следующим образом.

В **случае паралогов** общий граф <u>определён</u> в K.Yu. Gorbunov, V.A. Lyubetsky. Multiplicatively exact algorithms for transformation and reconstruction of directed path-cycle graphs with repeated edges. Mathematics, Oct 14 2021, Vol. 9, No. 20, Art. 2576.

Для пары структур <***a,b***> **общим** называется ребро, присутствующее в ***a*** и в ***b***. Иначе ребро называется **специальным**.

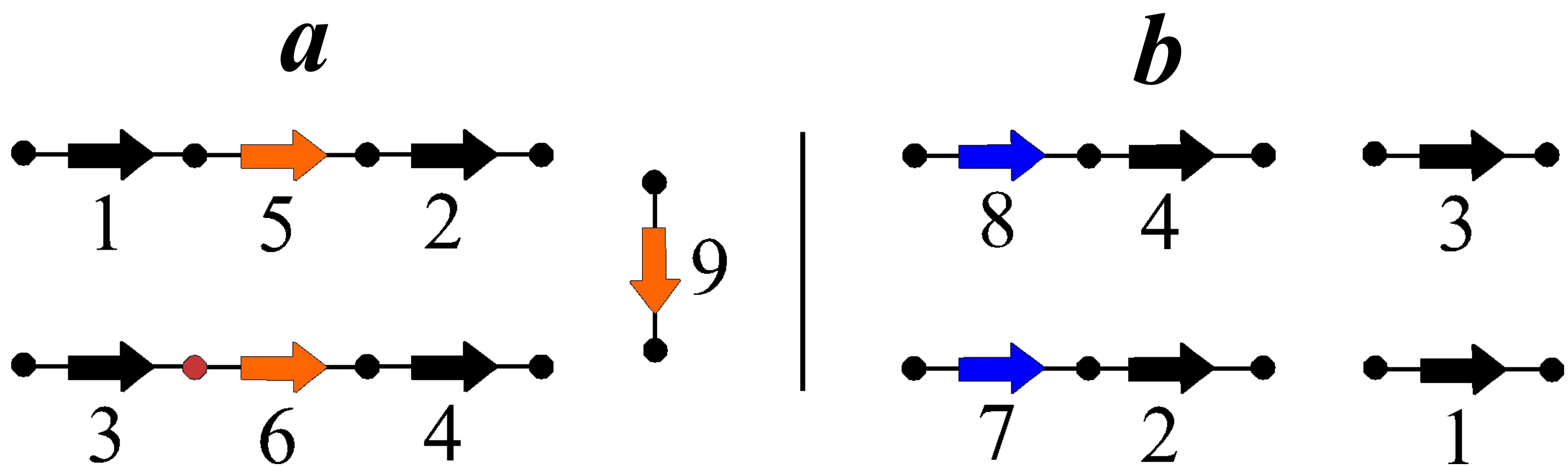
Для пары <***a,b***> **блоком** называется максимальный связный участок, состоящий из только рёбер в ***a*** или только рёбер в ***b***, т.е. только из специальных рёбер.

Край блока <u>*помечаем* именем блока: его начало и конец не различаются</u>, на рисунках они – полужирная точка.

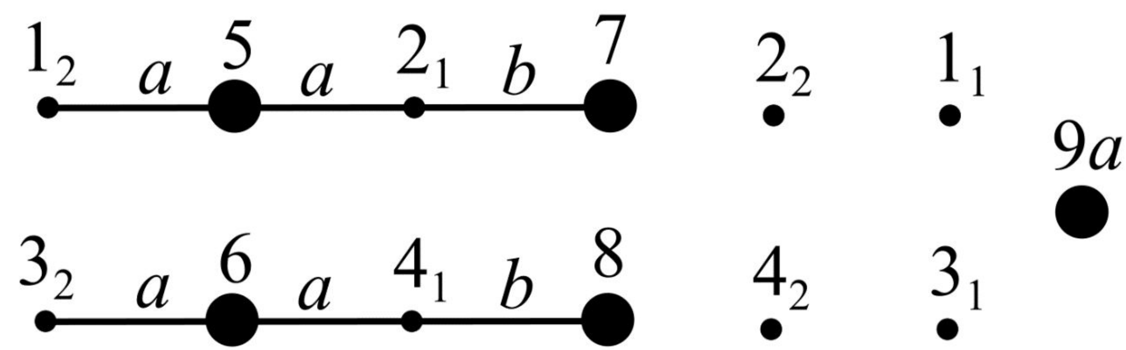
Как получить **общий граф *G*=*a*+*b*** по **структурам *a*** и ***b*** ?

Сначала примеры 1-2 такого перехода <***a***,***b***> → ***G*** .

Пример 1. Переход от <*a*,*b*> → *a*+*b*: склейки в *a* и *b* суть рёбра в *a*+*b*, а несклеенные вершины суть изолированные в *a*+*b*.



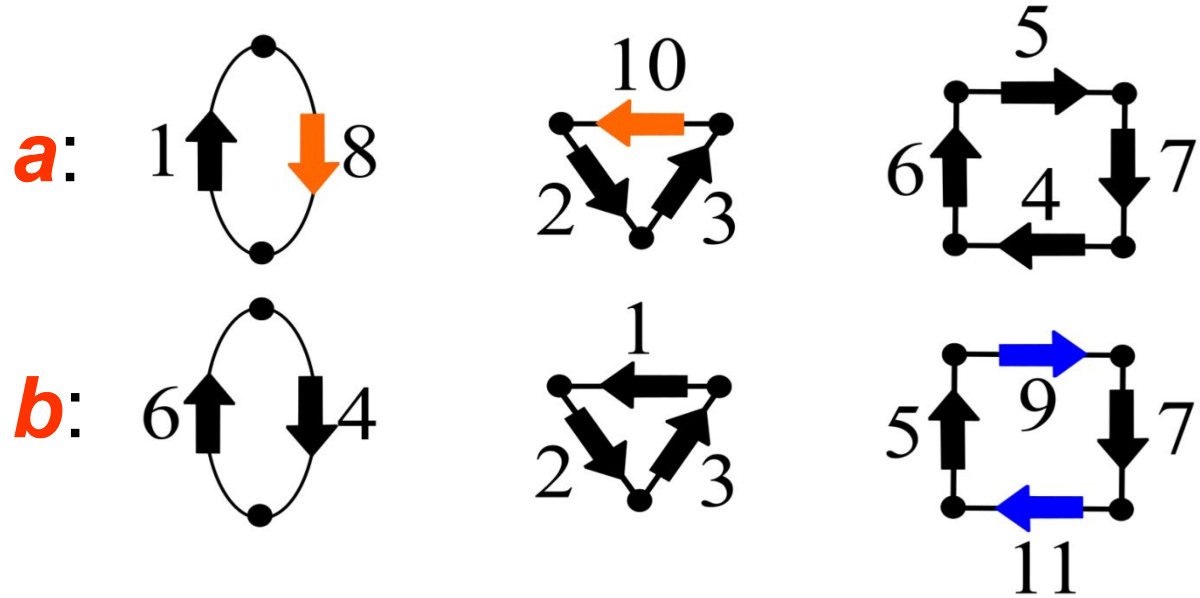
граф *G*=*a*+*b*:



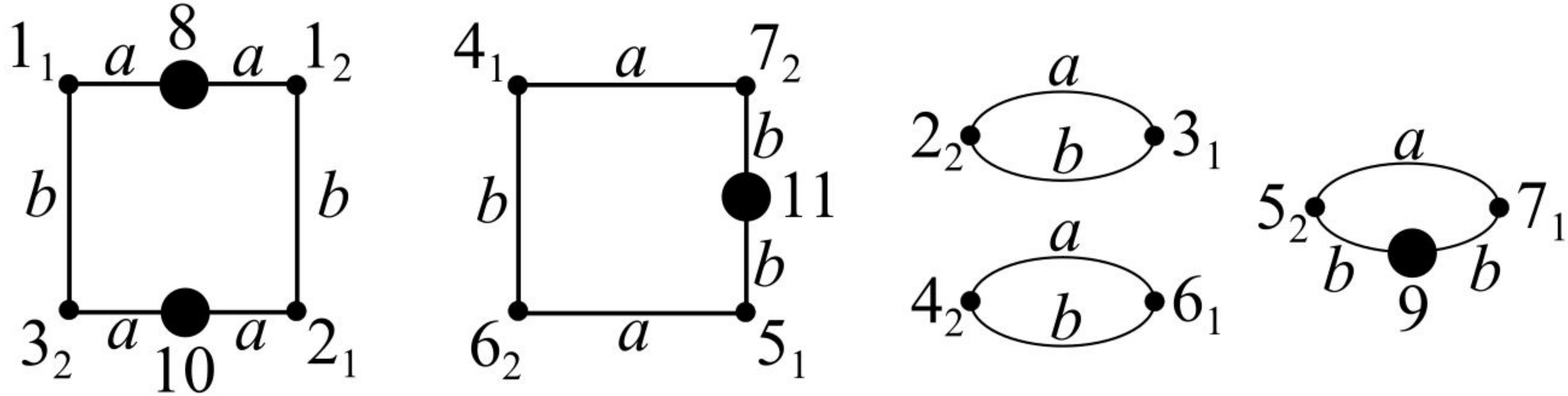
В *a* и *b* специальные рёбра помечены цветом, в *G*=*a*+*b* их краям соответствуют жирные точки.

Для любых *a* и *b* граф *G* состоит из циклов и путей, включая изолированные вершины.

Пример 2 : переход от <*a*,*b*> → *G*=*a*+*b*.



G=*a*+*b*:



Как получить **общий граф** $G=a+b$ **по структурам** a и b ?

Т.е. приведём определение a**+**b :

вершины в G – все края рёбер и блоков в a и b

(у блока один край!),

ребра в G – склейки в a и в b

(кроме склеек внутри блоков).

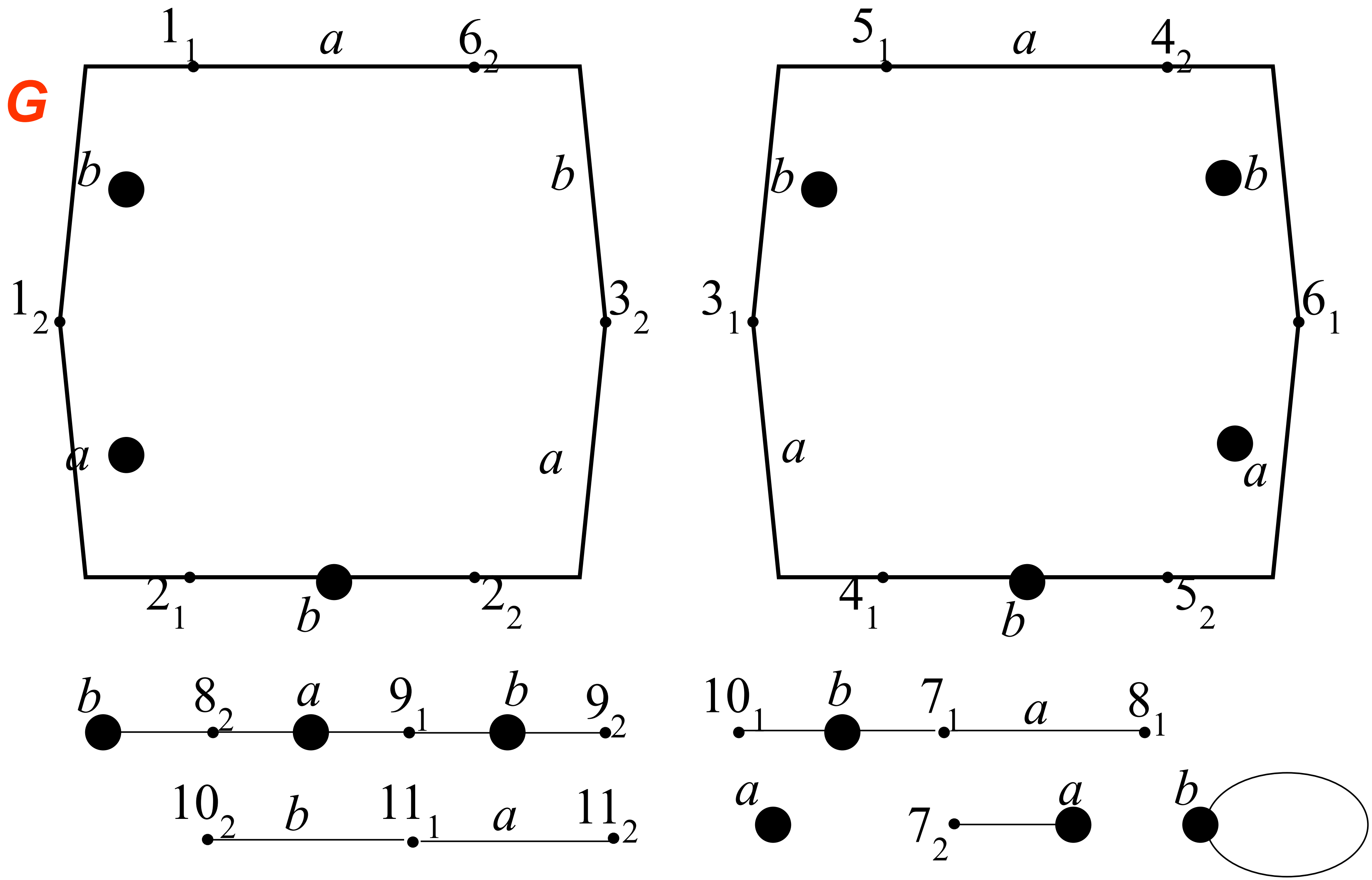
Изолированный блок изображается петлёй –

имя приписано петле и вершине!

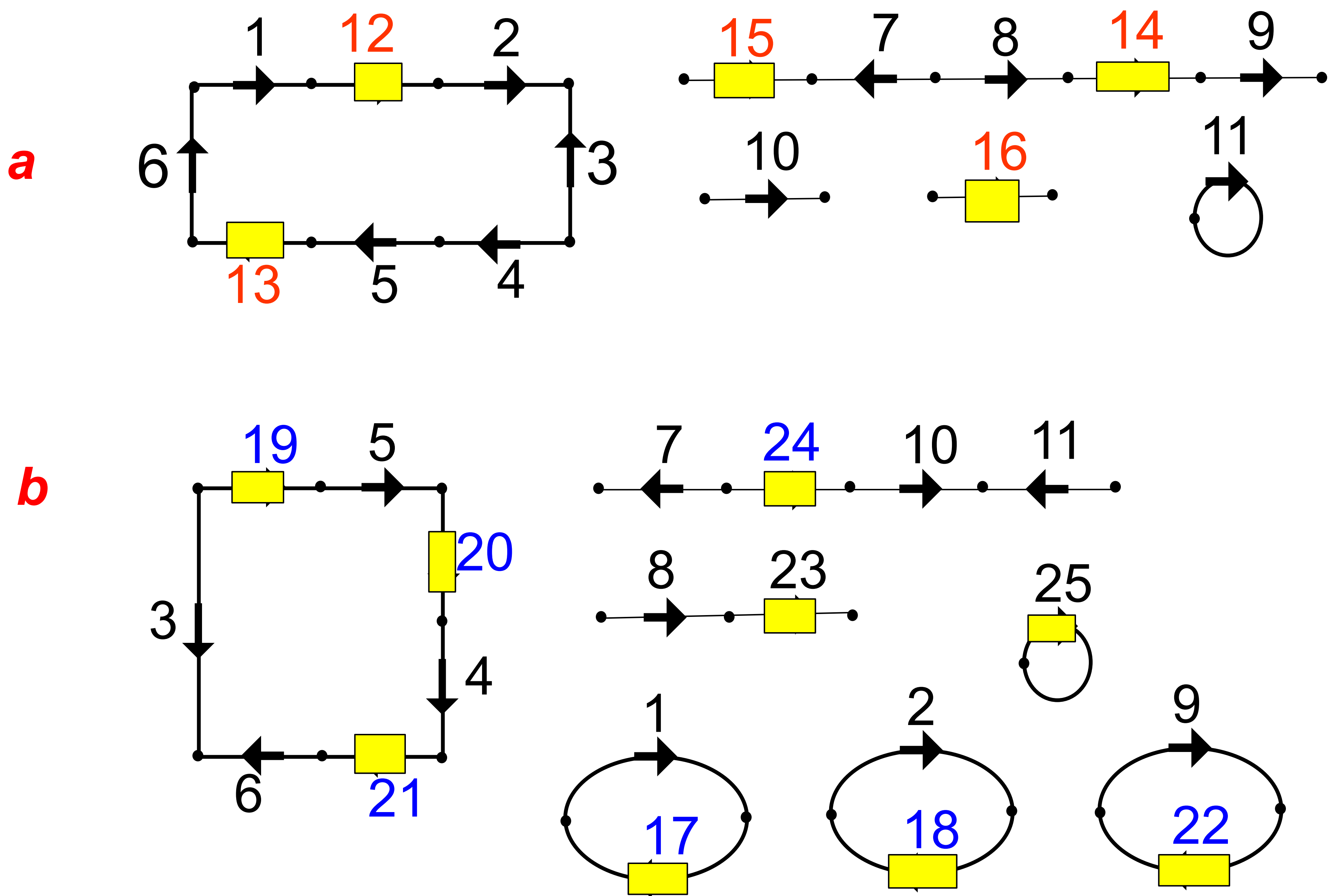
Графы a и b однозначно восстанавливаются по G.

Как? Сформулируйте правило восстановления.

Укажите структуры *a* и *b*, из которых получен следующий общий граф *G*. (Например, $1_2 b 1_1$ это цикл с ребром 1 и блоком 17.)



Ответ: *a*+*b* ⇨ <*a*,*b*>. Что тут не восстановилось?



В $a+b$ могут быть висячие рёбра, петли, изолированные вершины?

Ответ:

В примере 1, если ребра 1 нет, то ребро 5 в a станет в G висячей вершиной a и висячим ребром a (его другой край 2_1).

Часть определения $a+b$: циклы целиком из специальных рёбер в G суть **петли с пометкой вершиной**. Цепи целиком из специальных рёбер в G суть **помеченные изолированные вершины**.

(Изолированная вершина = точка.)

Вершину внутри *aa* или *bb*, и помеченную вершину (цепь длины =0 или вершину петли) назовём ***сингулярной***; остальные вершины назовём ***обычными***.

Ребро, хотя бы один край которого сингулярный, назовём ***сингулярным***; другие рёбра, у которых оба края обычные, назовём ***обычными***.

Заметим: в компоненте графа ***G*** имена ***a*** и ***b*** чередуются (если в ней ***a*** и ***b***, и «двойные» рёбра ***aa*** и ***bb*** считать за одно). Докажите.

Итак, ***графы G*** состоят из циклов, цепей, изолированных вершин, у которых каждому **ребру** приписано имя ***a*** или ***b***;

некоторым **краям** цепей (включая некоторые **изолированные вершины**) и **вершине петли** приписано одно из имён ***a*** или ***b***;

Разметка с тремя подряд одинаковыми именами запрещена. □

Край цепи, помеченный ***a*** или ***b***, и само ребро этого края называют ***висячим***. □

Висячие ребра играют в дальнейшем важную роль.

Отрезком в ***G*** назовём <u>максимальный</u> по включению связный участок из обычных рёбер.

В зависимости от длины он чётный или нечётный.

Размером графа G назовём число обычных рёбер в нем **+** половина числа <u>сингулярных невисячих</u> и <u>непетельных</u> рёбер **–** число <u>изолированных сингулярных</u> вершин;

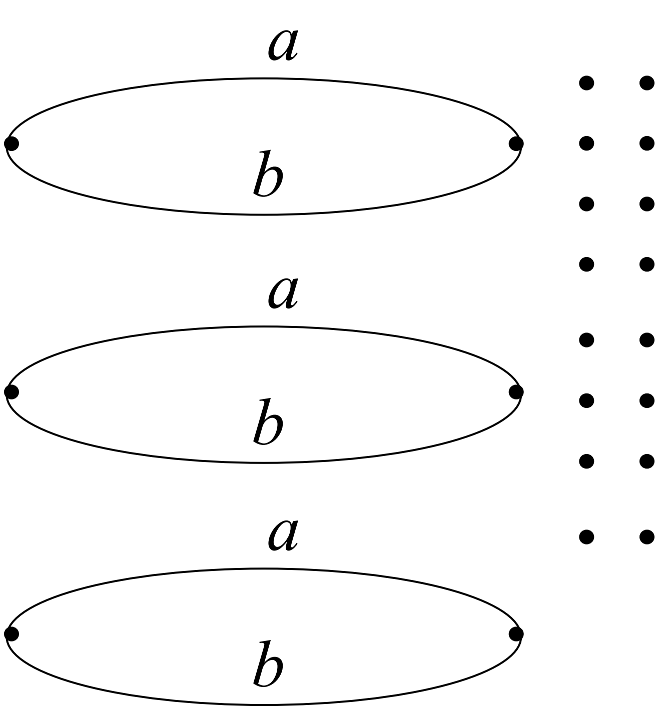
размер сингулярной изолированной вершины равен –1;

размер изолированной вершины и петли равен 0.

У отрезка длина и размер совпадают. □

У графа *G* на слайде 17 размер =17. Проверьте!

Определение 1. ***Финальным графом*** (=графом ***финального вида***) называется **общий граф *G***, состоящий из обычных рёбер <u>циклов длины 2</u>, помеченных буквами =именами ***a*** и ***b***, и ещё из <u>изолированных непомеченных вершин</u>:



На общие графы ***G*** со структур переносятся операции (см. след слайд), для которых определяются свои цены.

ЗАДАЧА ПРИВЕДЕНИЯ – найти минимальное приведение данного ***G*** к финальному виду.

Операции над парами структур переносятся на ***a*+*b*** и наоборот **по коммутативности**:

$$\begin{tikzcd} \langle a,b \rangle \arrow[r, "+"] \arrow[d, "o"'] & a+b \arrow[d, "o' =?"] \\ \langle o(a),b \rangle \arrow[r, "+"] & o'(a+b) = o(a)+b \end{tikzcd}$$

Эта диаграмма определяет связь операций в Задаче преобразования и операций в Задаче приведения, т.е. связь двух «математических структур» в <***a*,*b***> и в ***a*+*b***. Исходной является Задача преобразования и для неё операции естественные, а для Задачи приведения они производные и потому несколько корявые.

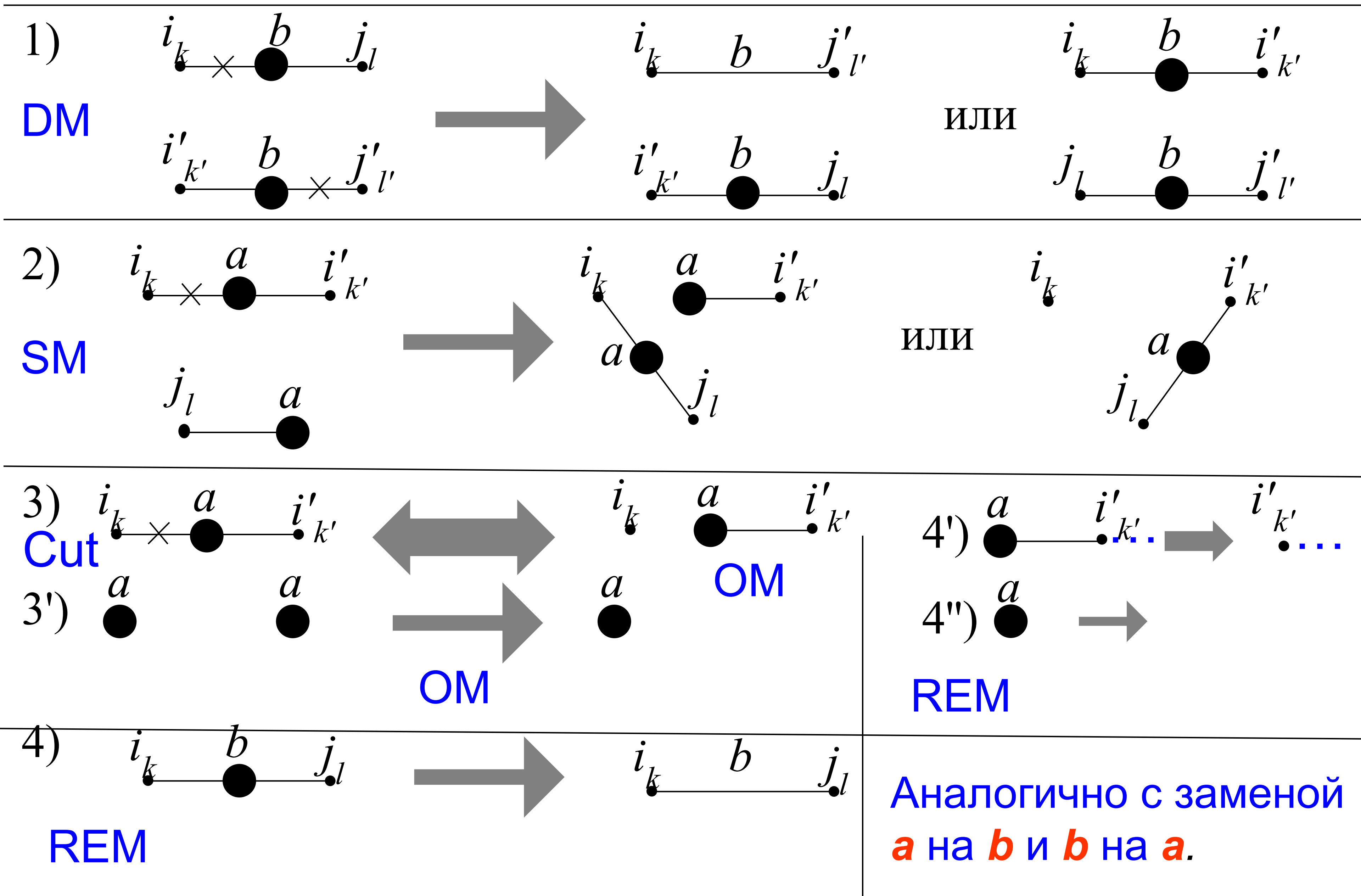
Лемма 0 (главная-1). Пусть цены DCJ-операций равны между собой, цены удаления сингулярных *a*- и *b*-вершин произвольные (обозначим их w_a и w_b, они равны ценам удаления участка в структуре ***a*** и вставки участка из структуры ***b***). Задача преобразования ***a*** в ***b***, эквивалентна Задаче приведения общего графа ***a+b***. Точнее, Кратчайшая цена равна! Минимальной цене приведения, и Кратчайшая последовательность преобразуется в Минимальную цепочку; и наоборот **линейным алгоритмом**. □

Фактически, здесь достаточно условия на цены:

DM = SM, OM+Cut ≥ SM ≥ max{OM,Cut}.

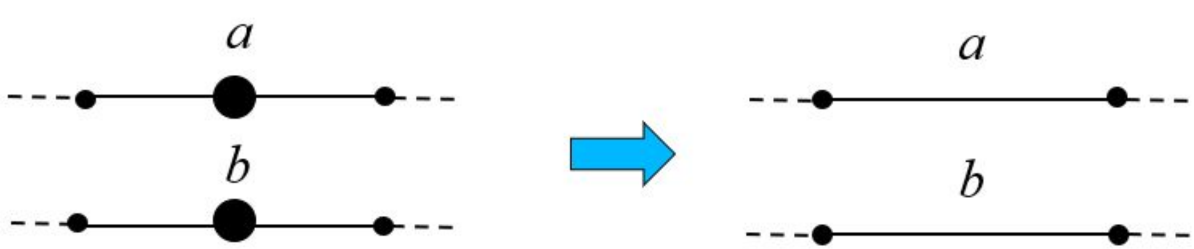
Это связь Задачи Преобразования и Задачи Приведения .

Операция над *G* называется одноимённо с операцией над структурой: как **DCJ**-операции и **удаление**. Они примерно такие:

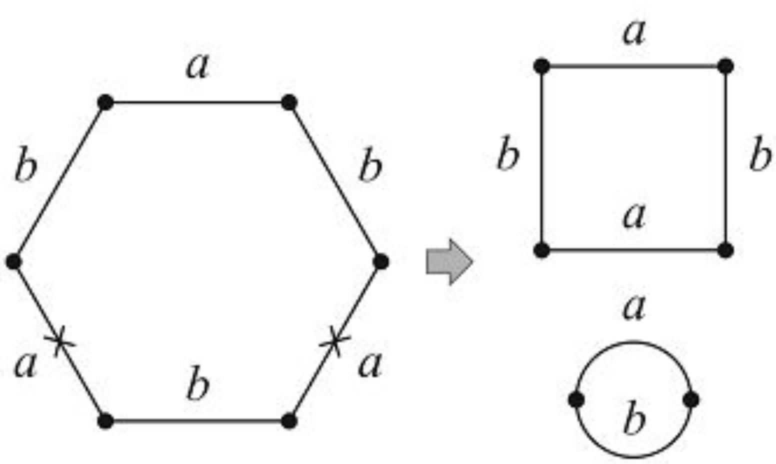


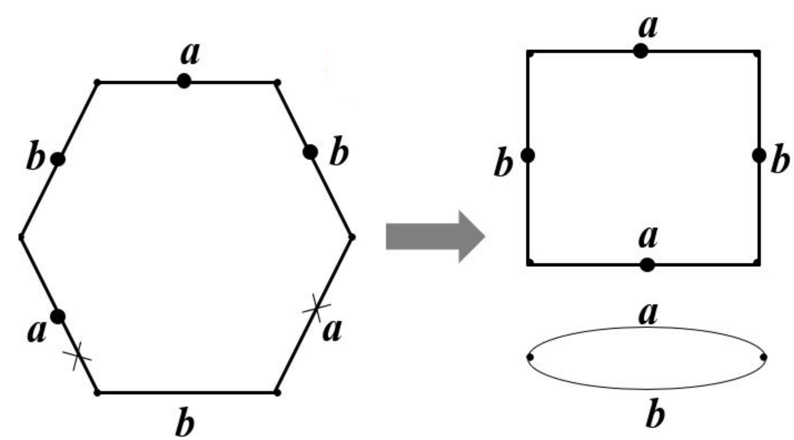
<u>Примеры применения этих</u> DCJ (= каждая из них называется **переклейкой**) и операции **удаления** к графу ***G* = *a*+*b*** :

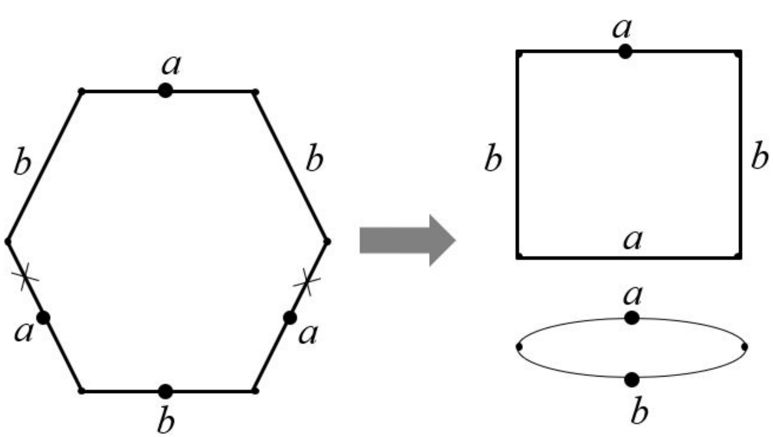
1) Rem: два идущих подряд ***aa*** или ***bb*** ребра заменить на одно ребро с тем же именем:



2) DM-переклейка: удалить два ребра с одинаковой пометкой и образовавшиеся свободные края соединить рёбрами с той же пометкой. Если образуется ребро с сингулярными концами, то стянуть его в сингулярную вершину (***объединение*** вершин):







Здесь DM для цикла – частный случай.

Точное определение! ***Одинарная склейка*** (**ОМ**): **добавление *a*-ребра между** свободными вершинами:

(1) обычными *b*-инцидентными вершинами; обычной *b*-инцидентной и *a*-висячей; обычной *b*-инцидентной и изолированной без пометки; обычной *b*-инцидентной и изолированной с *a*-пометкой;

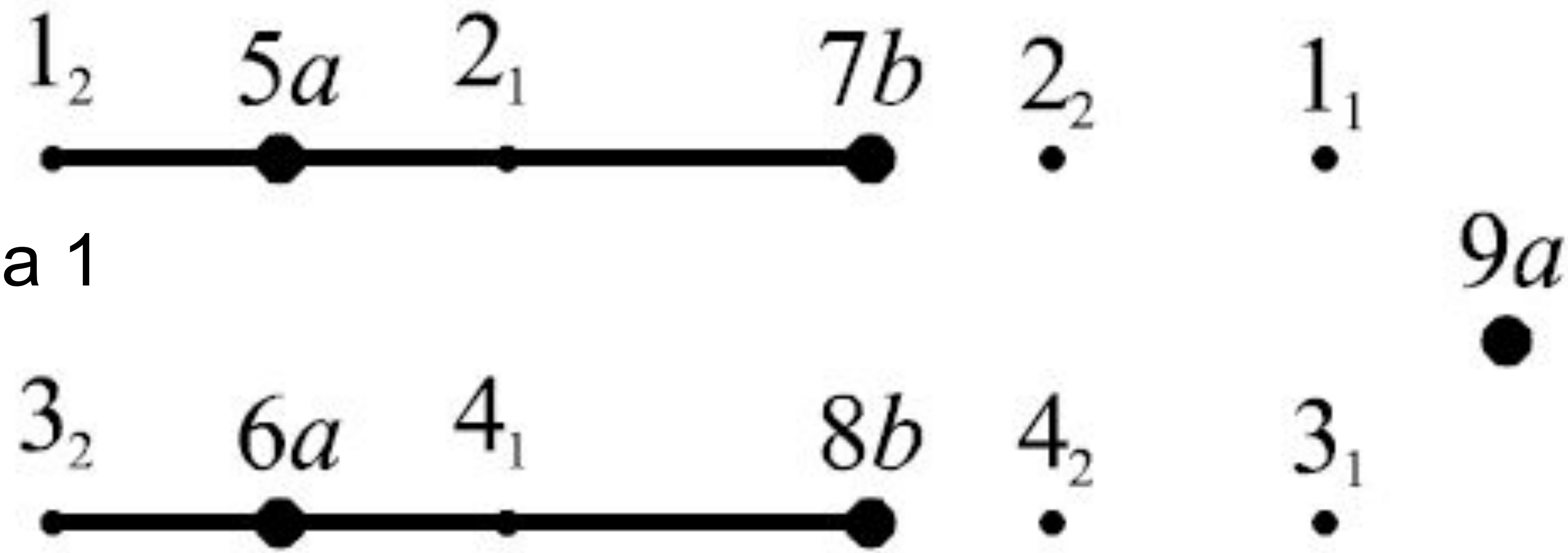
двумя изолированными, если обе не *b*-помечены;

(2) двумя *a*-висячими; **(3)** изолированной (обычной или с *a*-пометкой) и *a*-висячей. Аналогично с ***b*** вместо ***a***.

Задача: проверить это и ниже описания операций над ***a*+*b*** на коммутативность диаграммы. □

Отвлечёмся на пример приведения:

G
из
примера 1



Пусть цены всех операций равные. **Приведение** этого ***G***: ОМ-замкнуть в цикл каждую из двух цепей, затем выполнить 5 удалений сингулярных вершин (выпишите цепочку операций). Приведение **7-ю** операциями. Но используя SM можно обойтись **5-ю** операциями

(в этом проблема Задачи приведения). □

При разных ценах операций Задача приведения зн-о сложнее.

Точное определение! ***Полуторная переклейка*** (**SM**): удалить ребро и **добавить ребра с той же пометкой**, соединяющего **один из образовавшихся свободных краёв** со свободным:

1) обычным краем ребра с альтернативной пометкой (или изолированной без пометки), или свободной

2) висячей вершиной с той же пометкой или с сингулярной изолированной вершиной с той же пометкой.

3) Если SM применяется к крайнему *a*-ребру цепи, то как выше (если участвует край цепи); или (если участвует внутренняя вершина удалённого ребра) остаётся изолированная вершина и соединить *a*-ребром внутреннюю (*a*- или без пометки) вершину цепи с *b*-инцидентной или с *a*-висячей или с изолированной (*a*-помеченной или обычной). Аналогично с заменой ***b*** на ***a***.

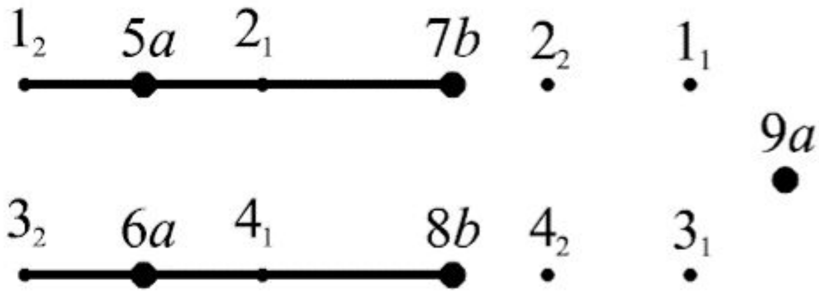
Во всех операциях, если образовались две смежные

одноимённые сингулярные вершины, то

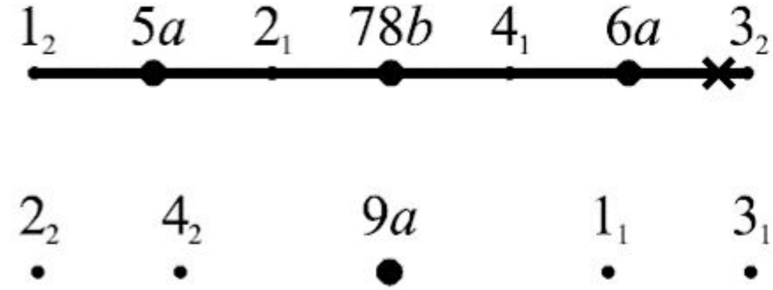
соединяющее их ребро стягивается в точку,

и эти две вершины ***объединяются*** в одну.

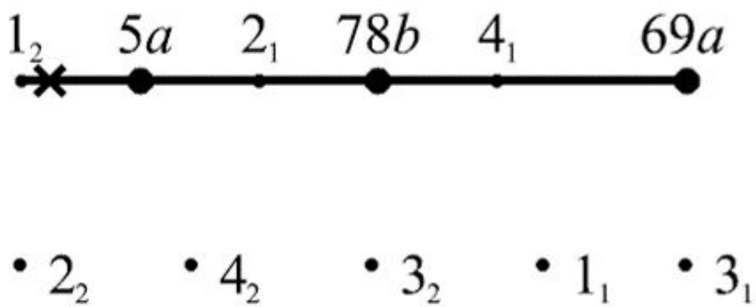
1) Общий граф *G*. ***OM*** для $7b$-$8b$



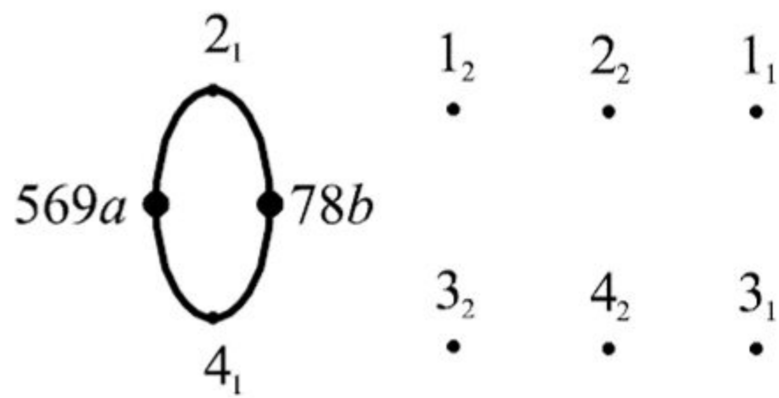
2) ***SM*** для крестика



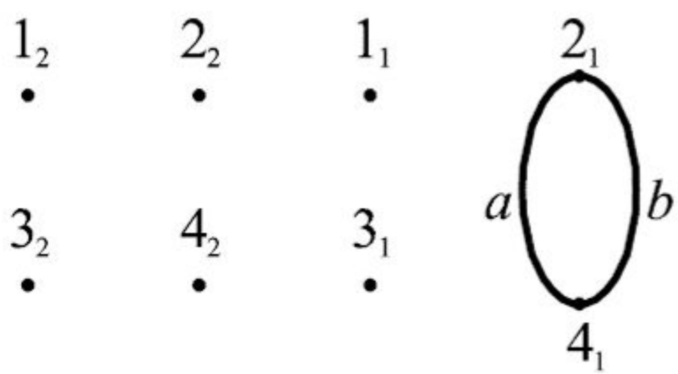
3) ***SM*** для крестика



4-5) ***a-Rem*** и ***b-Rem***



финальный граф для *G*



Для того же *G* из примера 1 приведение **5-ю** операциями, и это уже минимальное приведение !

Точное определение! ***Двойная переклейка*** (**DM**): на слайде 26

общий случай – удаляются два одноименных ребра и четыре

освободившихся края переклеиваются этими рёбрами.

Особые случаи: (Аналогично для ***b*** вместо ***a***.)

1) если *a*-петля и непетлевое-невисячее, то на месте удалённого: *aa*

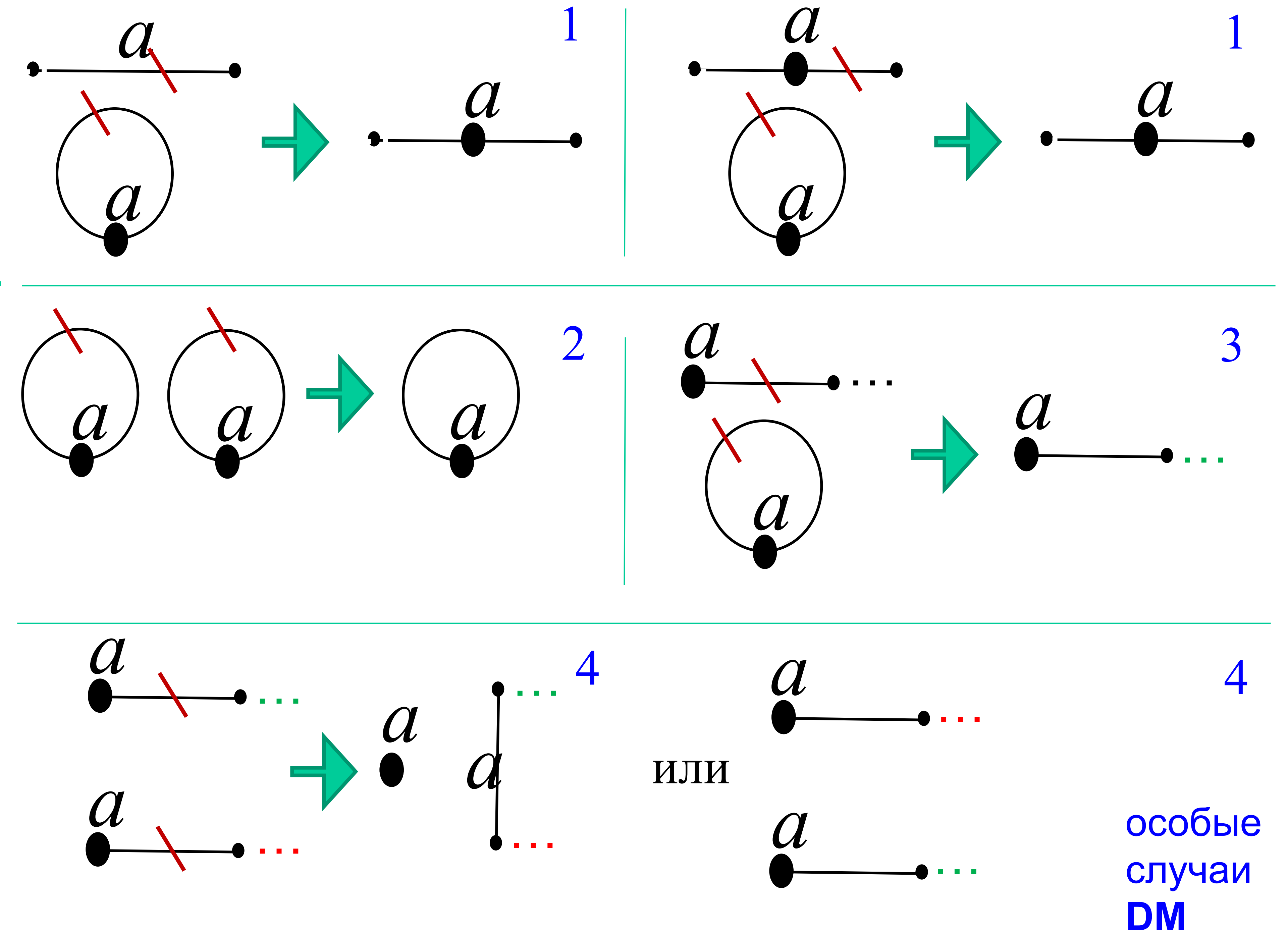
(если оно обычное) или то же самое (если сингулярное);

2) две *a*-петли дают одну *a*-петлю;

3) *a*-петля и *a*-висячая дают то же самое, но без исходной петли;

4) две *a*-висячих, результат отличается цифр пометкой или даёт *a*-

изолированную и *a*-ребро между цепями от обычных вершин.



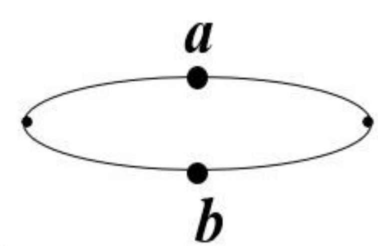
a
1
a
a
a
1
a
a
2
a
a
a
3
a
a
a
4
a
a
a
или
a
4
a
особые
случаи
DM

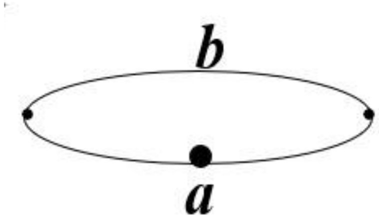
Предполагается (только) равенство цен DCJ-операций.

ТЕОРЕМЫ 1А. Построен линейный точный Алгоритм **приведения** любого графа ***G***. M

Ниже излагается **этот алгоритм** и доказываются его свойства. При этом важную роль играют свойства вспомогательного **Автономного алгоритма**.

Примитивным называется финальный вид или такой вид:





Автономным алгоритмом A (=***автономным приведением***) назовём последовательность операций над графом ***G*** :

ВЫРЕЗАТЬ все обычные рёбра;

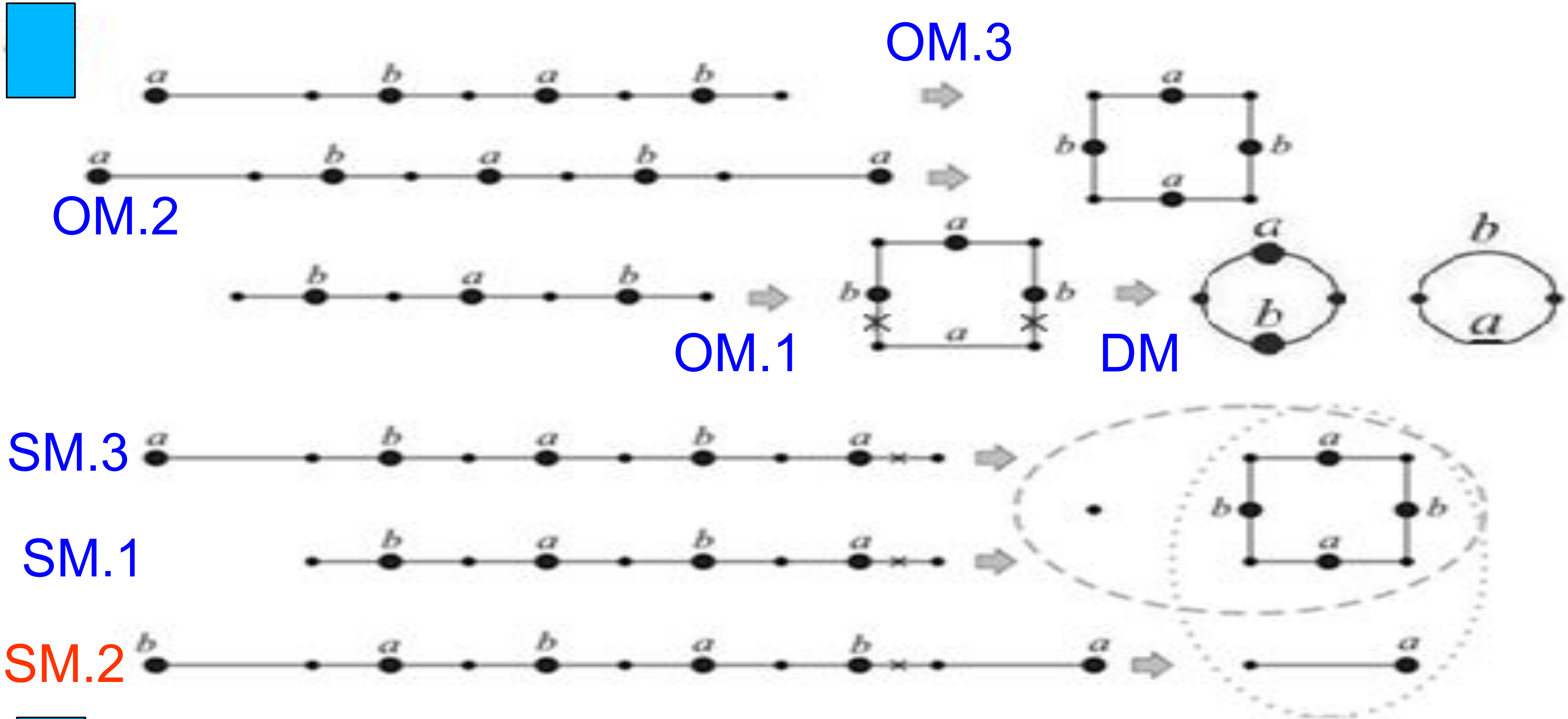
замкнуть цепи (размера >0) в циклы операцией OM или SM (если OM не позволяет замкнуть края цепи, то: SM – удаляем крайнее невисячее ребро, так что остаётся изолированная обычная вершина; иначе края цепи разноимённые висячие – удаляем 2-е с края *b*-ребро (если *b*-удаление дороже), так что остаётся висячее *a*-ребро; если образуется обычное ребро, то вырежем его);

измельчить все циклы до примитивных операцией DM, при которой из цикла вырезается цикл размера 2 с самой дешёвой сингулярной вершиной (если она дешевле цен DCJ);

удалить все сингулярные вершины и петли.

Автономной ценой A(G) графа ***G*** назовём суммарную цену цепочки его автономного приведения.

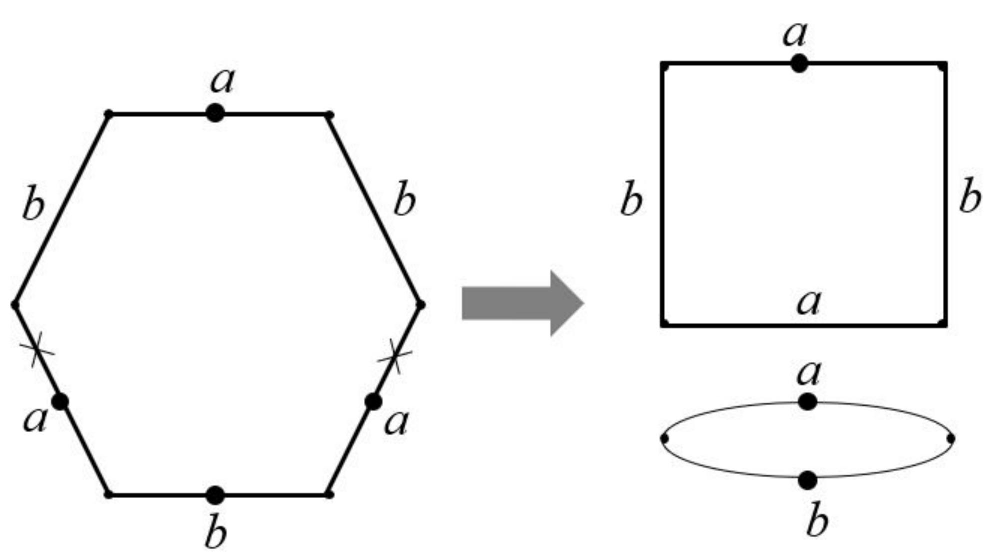
Как **ЗАМКНУТЬ цепь в цикл DCJ-операциями?** Ответ: (типы цепи сверху вниз: 1*b*, 2*b*, 3*b*, 1*a*, 3, 2 – все случаи!)



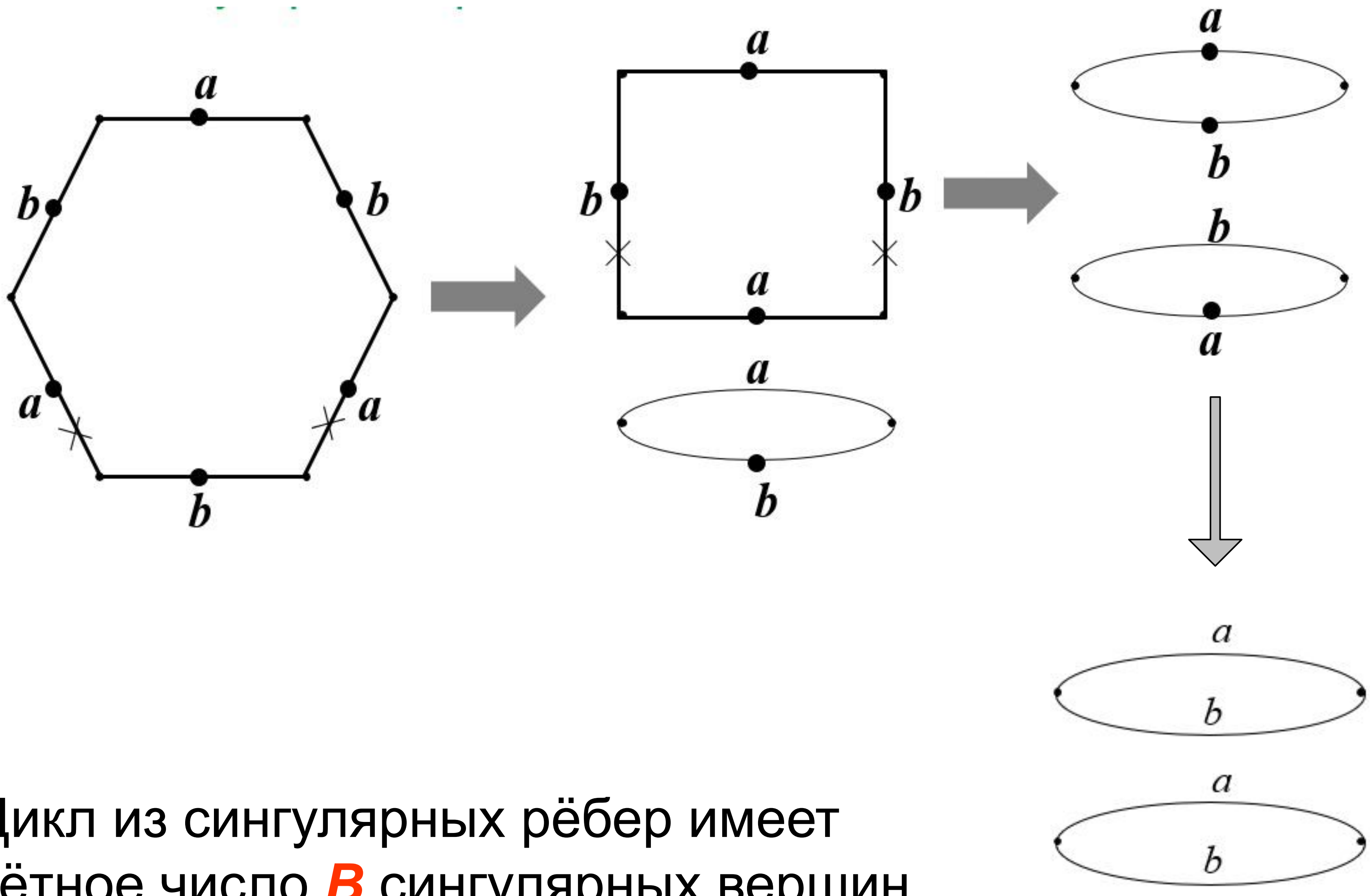
При замыкании обычное ребро может появиться!

Нап-р, при замыкании цепи aabbaa появляется обычное ребро b.

Тогда его нужно **вырезать**: в результате цепь не образуется и обычных рёбер уже нет:

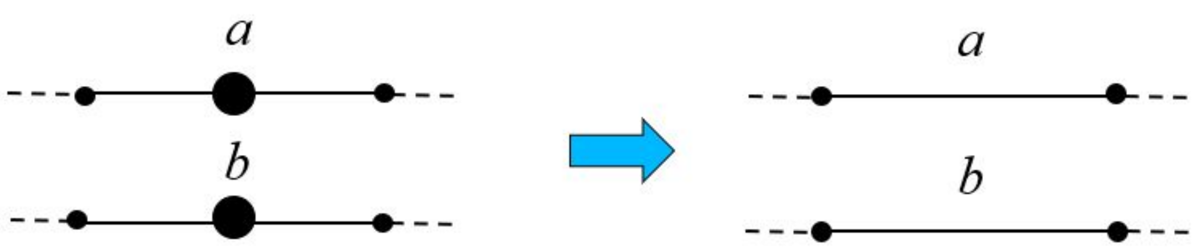


Как **ИЗМЕЛЬЧИТЬ** цикл ? (Затем **удалить** сингулярные вершины.)



Цикл из сингулярных рёбер имеет чётное число ***В*** сингулярных вершин.

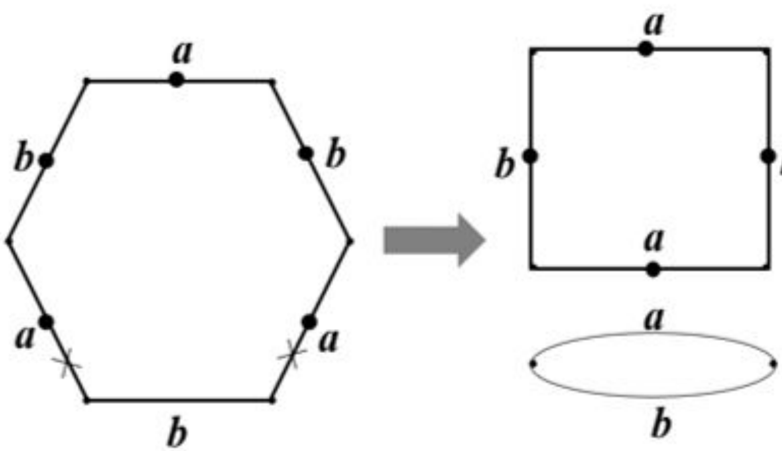
Удаление сингулярной вершины: два идущих подряд ребра ***aa*** или ***bb*** заменить одним ребром с тем же именем:

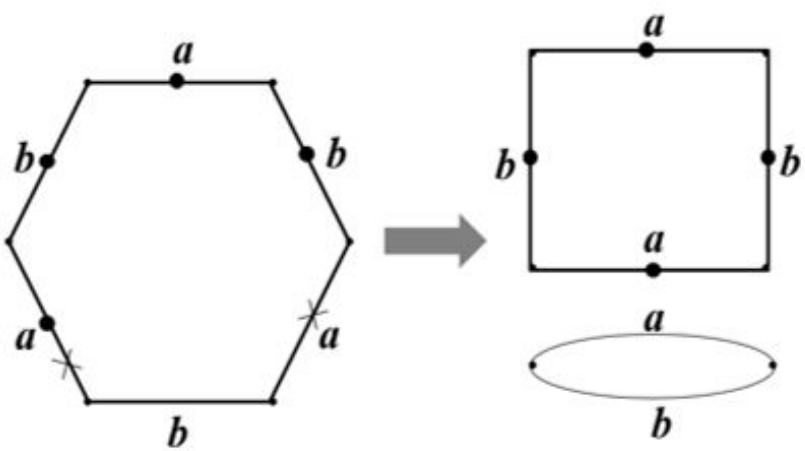


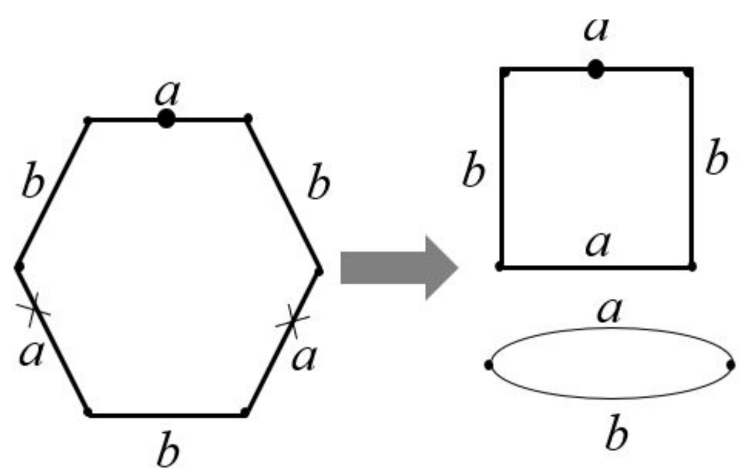
Висячая удаляется с её ребром; от изолированного ребра остаётся обычная вершина; от сингулярной вершины ничего не остаётся. См. слайд 26.

Если в *G* имеется какое-то обычное ребро *X*, то его следует **ВЫРЕЗАТЬ**. Это ОЗНАЧАЕТ применить операции переклейки к рёбрам соседним к *X*.

Примеры:







На следующем слайде показаны все случаи **ВЫРЕЗАНИЯ**: соседи вырезаемого ребра есть

<u>сингулярное невисячее и обычное</u>,

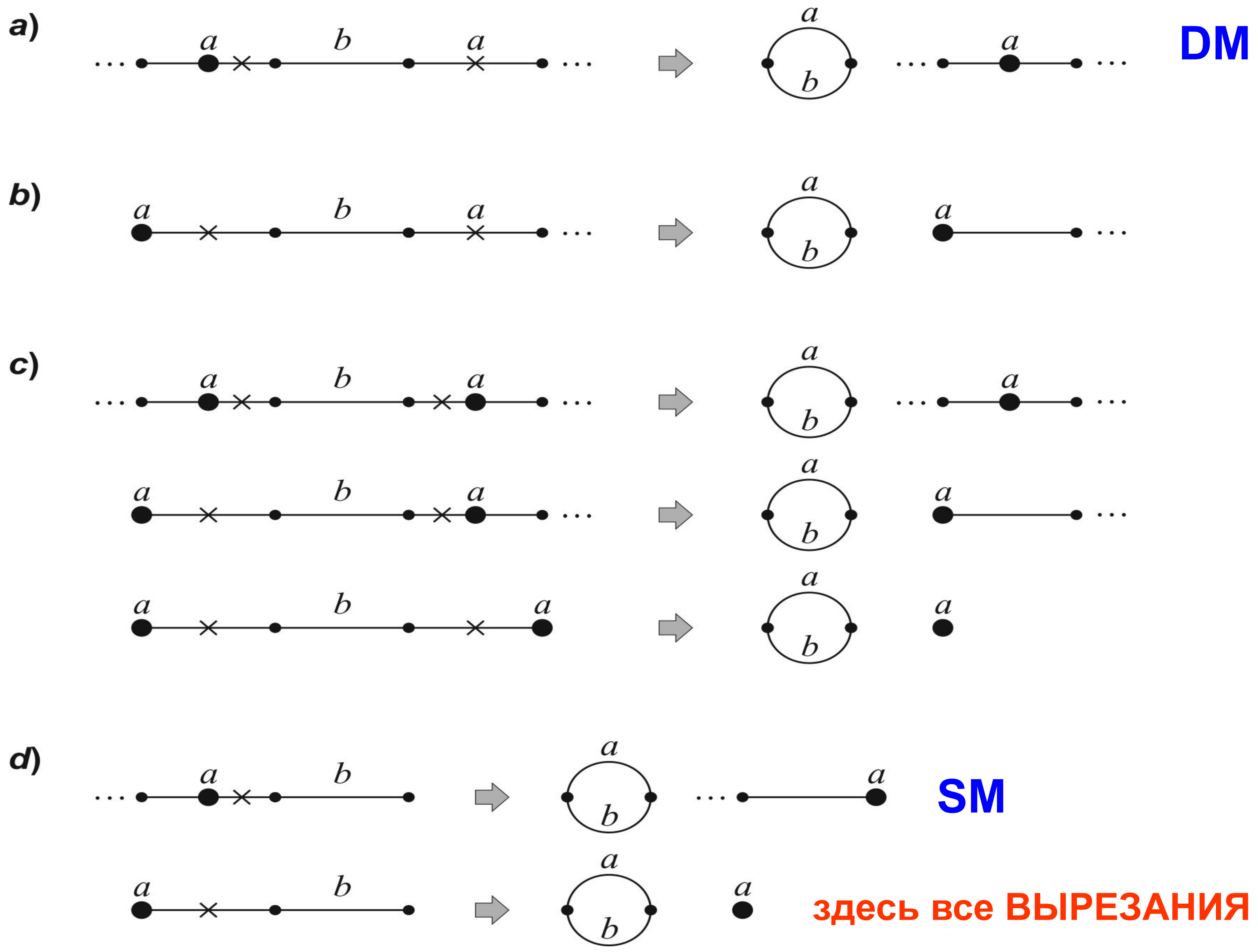
<u>сингулярное висячее и обычное</u>,

<u>оба сингулярные невисячие</u>,

<u>сингулярное и 1 висячее</u>,

<u>оба сингулярных висячих</u>,

<u>обычное крайнее и сингулярное невисячее</u> или <u>висячее</u>.

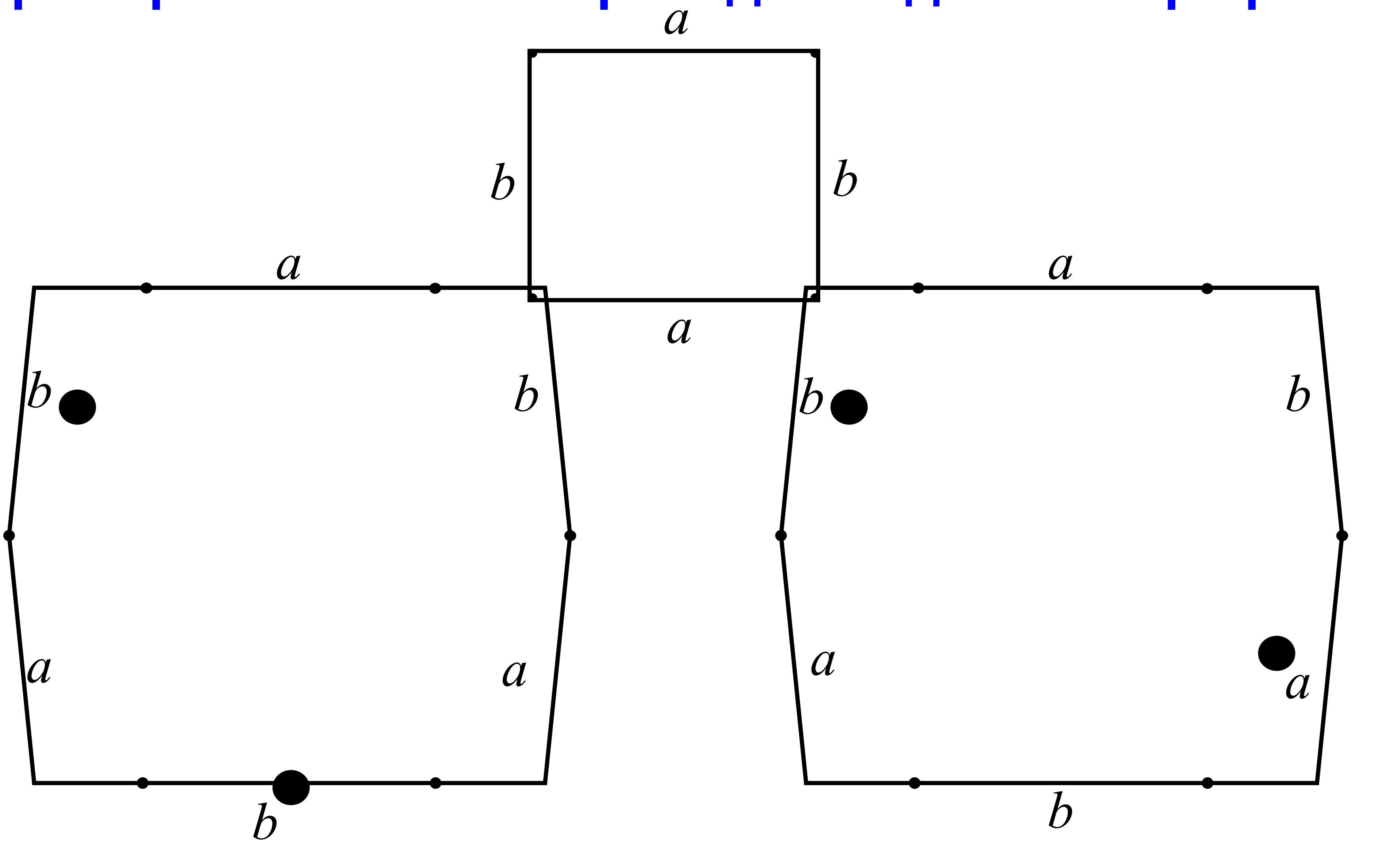

a)
b)
c)
d)
DM
SM
здесь все ВЫРЕЗАНИЯ

Обычные рёбра внутри цепи удаляются DM справа налево: остаётся одно ребро (если отрезок нечётный) или два ребра (если он чётный). В обоих случаях как выше.

Обычные рёбра с краю цепи удаляются **SM** с предкрая отрезка: если осталось 1обычное, то образуется висячее (как выше). 1изолированное-обычное с помощью **Cut**.

Отрезки удаляются в начале будущего алгоритма и больше не появляются, кроме обычного ребра внутри цепи.

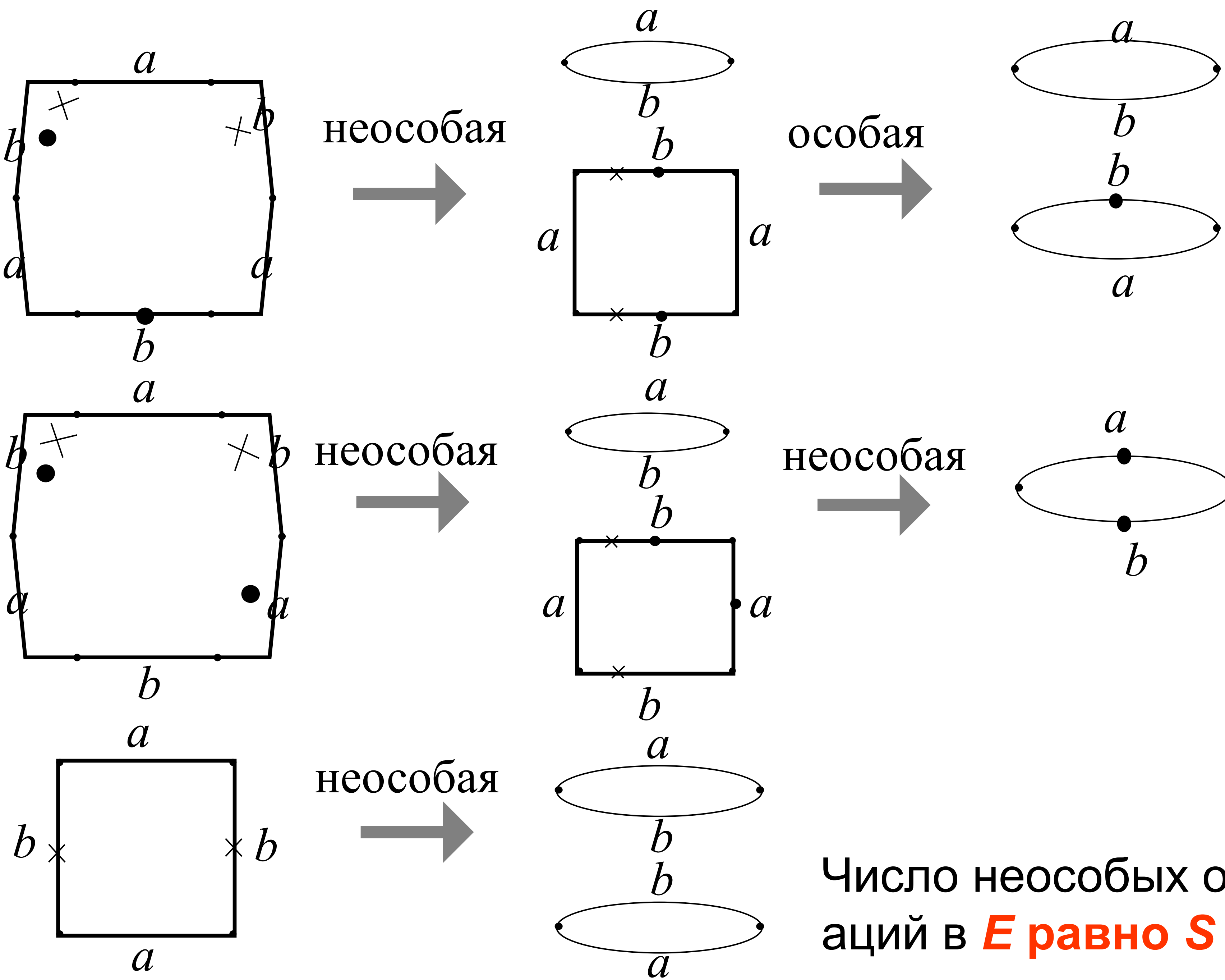
Пример автономного приведения данного графа *G*:



Определение 2а. Операция, которая <u>уменьшает число сингулярных вершин</u> в ***G***, называется **<u>ОСОБОЙ</u>**.

Иначе **НЕОСОБОЙ**.

Автономное приведение ***G***, получим цепочку ***E*** :



Число неособых опер-
аций в ***E* равно *S*** =4.

Определение 3.

СИСТЕМОЙ УЧАСТКОВ в приводящей цепочке ***E*** (от *G*) назовём любое множество связных участков, пересекающихся по последнему графу из каждого участка и покрывающих всю цепочку (<u>последний граф каждого участка является первым графом следующего участка</u>, кроме последнего участка):

$$E\text{: } G \to \ldots \to R \to \ldots \to S \to \ldots \to \textit{фг}$$

$$E\,\textit{фг}\; G \to \ldots \to \left[R \ldots \to R'\right] \ldots \ldots \to \ldots\ldots \left[S \to \ldots \to \quad\right]$$

Обозначим ***c*(*s*)** **сумму цен** всех операций в **участке** ***s***.

Определения 4. ***Приводящий алгоритм*** – это алг на графе *G*, который выдаёт приводящую цепочку для *G* <u>вместе с системой участков</u> в ней, и этот алгоритм тождественен на финальном графе. Пусть ***T*(*E*)=*T*(*G*,*E*) – сумма цен операций в *E***.

Качество *P*(*s*) *участка* *s* = *G* … *R* равно

$$P(s) \;=\; \underline{A(G) - A(R) - c(s)}.$$

<u>Суммарное качество</u> P(E) цепочки E равно

$$P(E) \;=\; \square_s\, P(s),$$ т.е.

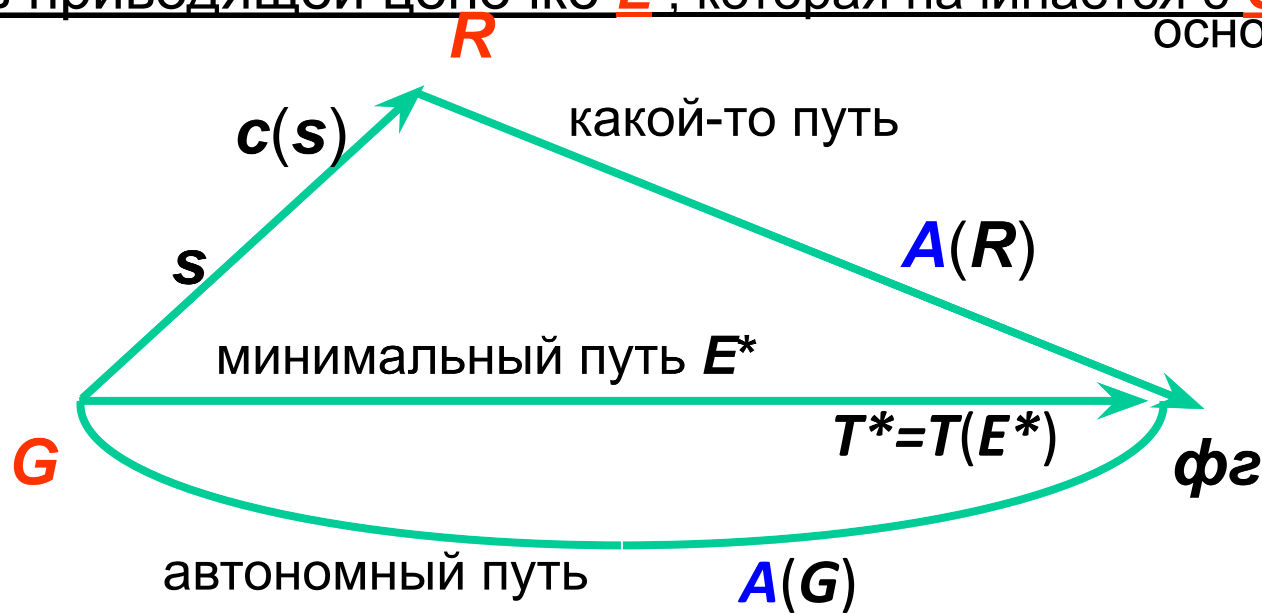
сумме ***качеств*** *P*(*s*) каждого участка *s*; *s* – <u>композиция операций</u>.

В следующей Лемме 2 *автономный алгоритм* ***A*** – любой фиксированный приводящий алгоритм, а *финальный граф* – любой фиксированный граф, не обязательно конкретного вида, определённого выше.

обозначим A(□) цепочку автономного приведения на данном аргуме-нте □, т.е. от G до ***фг*** и от R до ***фг***. И ещё выделен ***участок***

s **=** ***G* …** ***R*** <u>в приводящей цепочке *E* , которая начинается с *G*</u>.

основной слайд!



< **Если** $P(s) = \underline{A(G) - A(R) - c(s)} > 0$, т.е. $\underline{A(G) > A(R) + c(s)}$, то переход в состояние ***R*** <u>выгодно</u>, чтобы найти кратчайшее приведение ***G***. Т.е. **качество участка** ***P*(*s*) хорошее**, если ***P*(*s*)>0**, **и тем лучше, чем больше** ***P*(*s*)**.   Аналогично для $\underline{P(E) = □_s P(s)}$. >

Лемма 2. *Для <u>любой приводящей цепочки</u> E с системой участков, которая начинается с графа G= G(E) выполняется* $T(E) = A(G) - P(E) \to \min$.

Доказательство.

$$E\text{: } G \to \ldots \to R \to \ldots \to S \to \ldots \to фг$$

$$E\text{: } G \to \ldots \to [R \ldots \to R'] \ldots \ldots \to \ldots\ldots\ldots [S \to \ldots \to]$$

$$\boldsymbol{A(G) - T(E) = P(E), \; \square E} \;,$$

где, например, $\boldsymbol{P(G,s)} = \boldsymbol{A(G)} - \boldsymbol{A(s(G))} - \boldsymbol{c(s)}$ и $\boldsymbol{s}$ – первый участок в $\boldsymbol{E}$. Индукцией по числу участков в $\boldsymbol{E}$. □

Смысл Леммы 2: чтобы $T(E)$ (где $G=G(E)$ фиксировано – закреплённый конец) было минимальным $P(E)$ должно быть максимальным! Значит нужно подобрать такую цепочку E^* и систему участков в ней, чтобы $P(E^*)$ было максимальным по всем E для данного G.

Тогда $T(G,E^*)$ с **этим** E^* **будет минимальным**:

$$T(G,E^*) = A(G) - P(E^*) \ .$$

Итак, по G хотим выдать приводящую цепочку E^* с максимальным качеством, а система участков в E^* в ней может быть любой (=последовательность останется кратчайшей)!

Оказывается это можно сделать алгоритмически: линейно по сложности и точно по результату !!

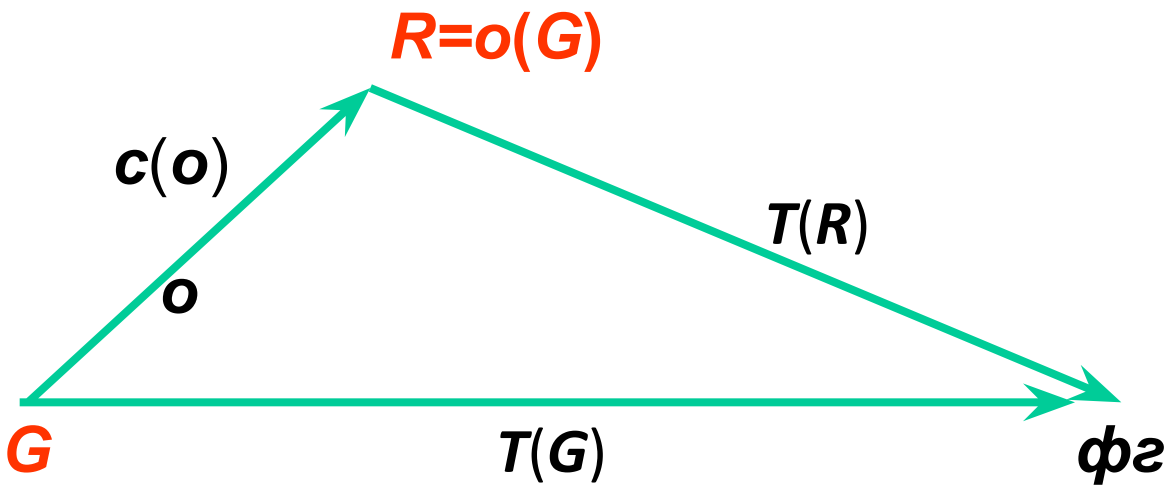
Определение 5. Пусть ***T*** любой приводящий алгоритм.

Неравенство треугольника *для* ***T*** означает:

для любой операции ***o*** *и любого графа* ***G*** *выполняется*

$$\underline{T(G) \le T(o(G)) + c(o)}, \qquad (*)$$

где o(G) – результат применения операции o к G.



Лемма 3. *Для любого приводящего алгоритма* ***T*** ***точность T*** *эквивалентна* ***неравенству треугольника*** *для* ***T*** . □

Эта лемма не связана с Леммой 2 .

Док-во. Можно считать цены операций – натур. числа; все их суммы – натур. числа. Предположим неравенство треугол. Выполним индукцию по величине $C(G)$ *минимальной суммарной цены* (по всем приводящим цепочкам от G). Базис: $C(G)=0 \Rightarrow T(G) \leq C(G)$ (приводящая цепочка пустая). Рассмотрим непустую минимальную приводящую цепочку от G, в которой первую операцию обозначим o. По предположению индукции $(C(o(G))<C(G)) \Rightarrow T(o(G)) \leq C(o(G))$.

По нерав. треугол. $T(G)\leq c(o)+T(o(G))$ и $\leq c(o)+ C(o(G)) = C(G)$ (так как любой остаток минимальной цепочки является минимальной – докажите). Итак, $T(G) = C(G)$.

Ещё проще в обратную сторону. □

Лемма 1. *Пусть w_a и w_b – цены удаления сингулярных a- и b-вершин, $w_a \le w_b$, цены DCJ операций равны* 1. *Тогда* ***автономная цена A(G) графа G равна***

$$A(G) = (1-w_a)\cdot(0.5d+0.5f-c) + w_a\cdot(B+S+D) + (w_b-w_a)\cdot K_b. \quad (1)$$

Здесь в ***G***: ***d*** – суммарный ***размер G*** (размер всех компонент), ***B*** – число сингулярных вершин; ***f*** – число нечётных (по размеру) цепей, ***c*** – число циклов (не петель);

S – сумма целых частей половин длин отрезков, сложенная с числом крайних (на цепи) нечётных отрезков, и минус число циклических отрезков , т.е. $S = (\sum_{\gamma}[l_\gamma / 2]) + c_0 - c_1$

, где c_0 число крайних нечёт отрезков и c_1 число цикл отр-ов;

K_b – число компонент, содержащих *b*-сингулярную вершину;

ещё характеристике графа ***G***:

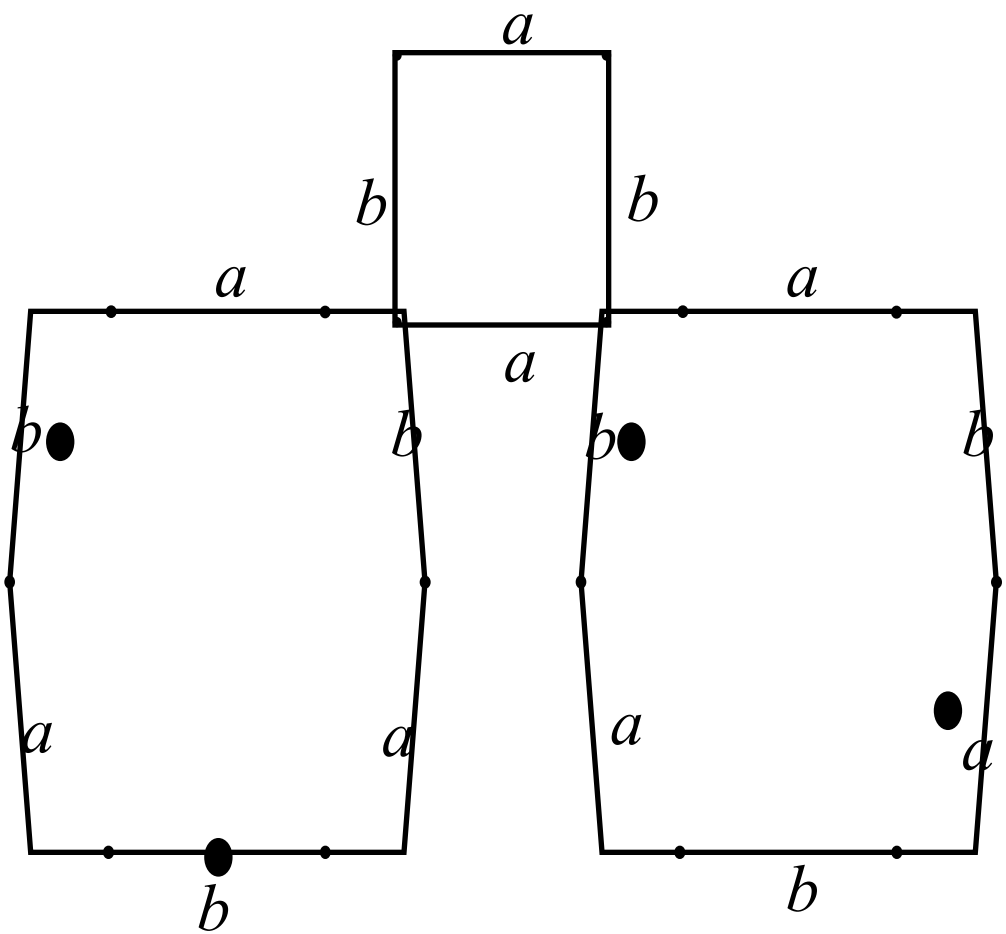
D – число цепей, которые замыкаются в цикл неособой операцией (при автономном приведении).

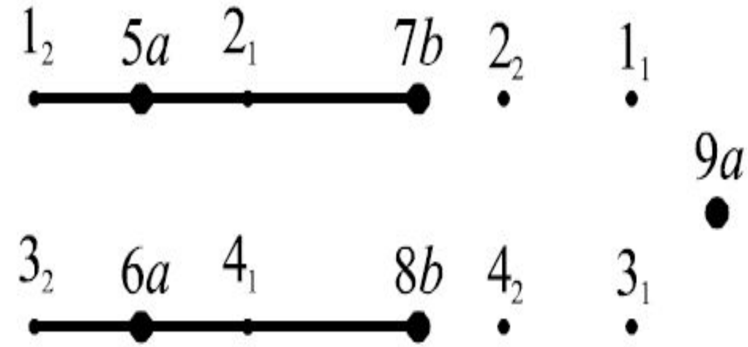
< Лемма. Это ***D*** **=** числу цепей типов **1*a*, 1*b*, 3*a*, 3*b*,** 3. >

Докажите.

Можно: cost(DCJ)=1, $w_a \le w_b$ или $w_b \le w_a$ (тогда *a* и *b* меняем).

Пример к лемме 1, **автономное приведение графа *G***:





$$A(G) = (1-w_a)\cdot(0.5\cdot 17+0.5\cdot 3-3) + w_a\cdot(9+4+2)+(w_b-w_a)\cdot 4 = 7+4w_a+4w_b.$$

Для циклической части, выполним 5 DM-переклеек, одно *a*-удаление и два *b*-удаления – цена $5+w_a+2w_b$. Для линейной части замкнём каждую из двух цепей в цикл (2 OM-переклей-ки) и удалим 3 *a*-вершины и 2 *b*-вершины – цена $2+3w_a+2w_b$.

По леммам 1 и 2 получим!!:

для любой приводящей цепочки ***E*** с системой участков*,*

начинающейся с графа ***G***= *G*(*E*), *выполняется*

$$\boldsymbol{T(E) =}$$

$$\underline{(1-w_a)\cdot(0.5d+0.5f-c) + w_a\cdot(B+S+D) + (w_b-w_a)\cdot K_b} - P(E) \qquad \textbf{(2)}$$

В правой части находятся характеристики закреплённого

конца – только исходного графа ***G***, и ещё ***P*(*E*) –**

характеристика всей цепочки *E*.

Доказательство Леммы 1 будет позже.

Итак, ещё раз!: чтобы **min*T*(*E*)**, **нужно! max*P*(*E*).**

Проблема 1: ослабить предположение о равенстве цен DCJ-операций в Лемме 0 (главная1).

Перейти к рекурсивным счётным последовательностям с оракулом типов цепей вместо структур.

Для описания Алгоритма M

с произвольными ценами операций нужно важное

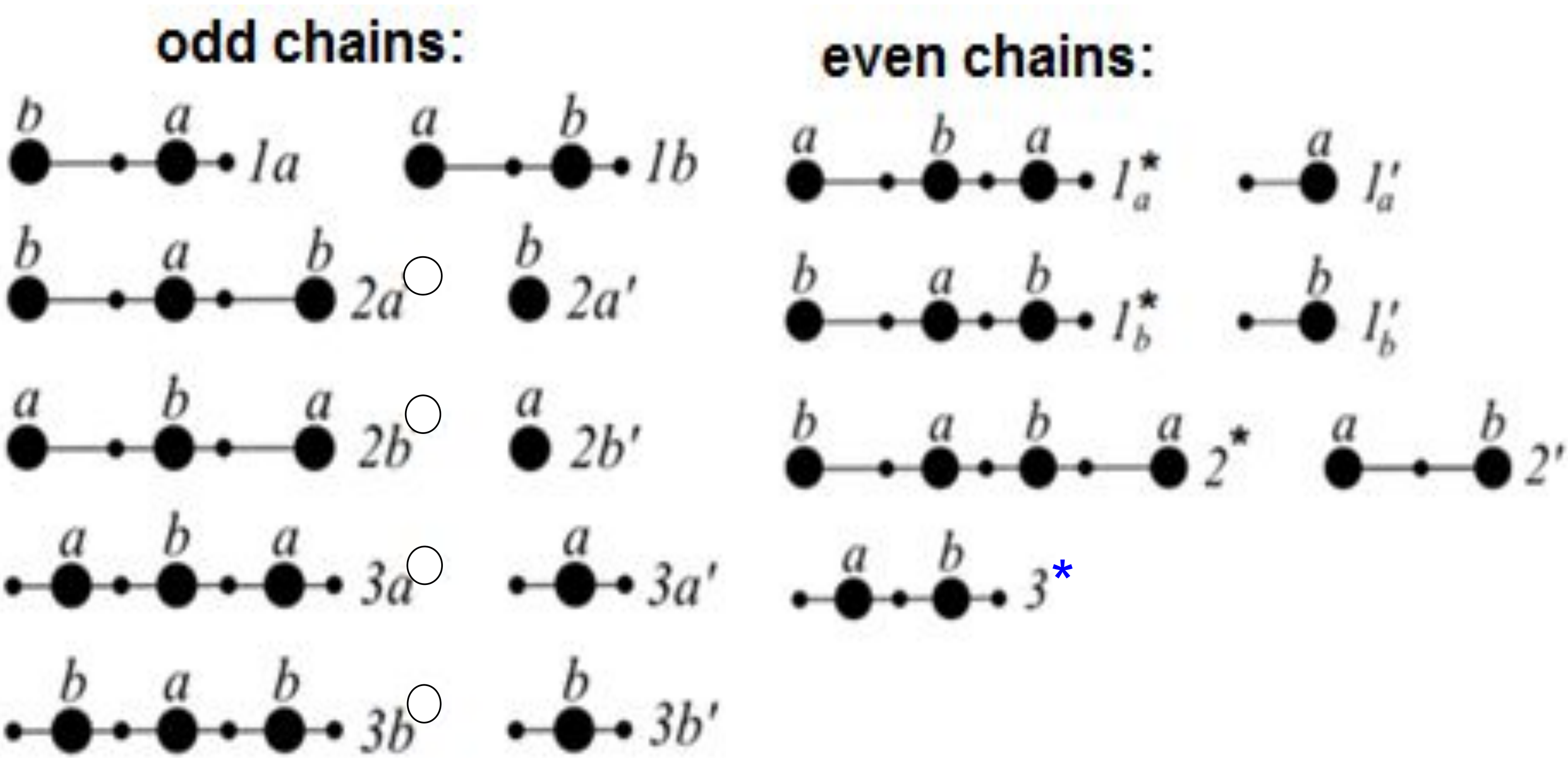
Определение типа цепи,

которое неявно встречалось выше много раз.

ТИПЫ ЦЕПЕЙ: с сингулярными вершинами и

ровно 1висячая, 2 висячих, 0висячих.

Или только обычными вершинами. (Штрих – предельный случай.)



(**0** – цепь без сингулярных вершин)

Определение 5 ***типа цепи*** (если цепь *не содержит* *обычных рёбер*): 1*a* – нечётная цепь с одним висячим *b*-ребром; 2*a* – нечётная цепь с двумя висячими *b*-рёбрами; 2*a′* – *b*-сингулярная изолированная вершина; **2*a*** – тип 2*a* или 2*a′*. Тип 3*a* – нечётная цепь без висячих рёбер с двумя крайними *a*-рёбрами, имеющая *b*-сингулярную вершину; 3*a′* – цепь *aa*; **3*a*** –тип 3*a* или 3*a′*. // Тип 1_a* – чётная цепь с одним висячим *a*-ребром, имеющая *b*-сингулярную вершину; $1_a{}'$ – висячее *a*-ребро; **1_a** – тип 1_a* или $1_a{}'$. Аналогично с заменой *a* на *b*. Тип 2* – чётная цепь с двумя висячими неинцидентными друг другу рёбрами; 2′ – два висячих рёбра, инцидентных общей обычной вершине; **2** – тип 2* или 2′. Тип 3* см. дальше.

Определение 5 ***типа цепи*** (если без обычных рёбер):

тип 3* – чётная цепь без висячих рёбер, но с сингулярными вершинами. // **0** – цепь без сингулярных вершин (с обычными рёбрами) или обычная изолированная вершина.

Тип цепи с *обычными рёбрами* определяется как тип цепи, полученной после их вырезаний,

что не зависит от последовательности вырезаний [4, лемма 6].

Фактически: тип цепи с *обычными* рёбрами, если они внутри цепи, одинаков до и после удаления, так как он определяется краями цепи и чётностью отрезка, которые сохраняются.

А, если *обычные* рёбра на краю цепи, то:
если отрезок нечётный, то после удаления возникает висячее ребро, иначе не возникает.

Операции **SM на конкретных типах**: $1a+1b=1_b{}^*$, $3a+2b=1_a{}^*$,

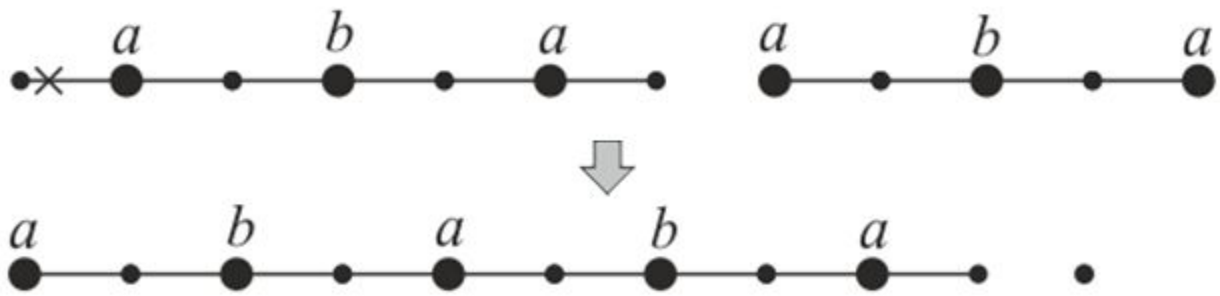
$3a+2b' = 1_a{}^*$, $3a'+2b= 1_a{}^*$, $3a'+2b'= 1_a{}'$, $3b+2a= 1_{\cancel{b}}$; $3^*+2 = 1_b{}^*$.

Разрезаемая цепь указывается первой.

Рисунки четырёх из них:

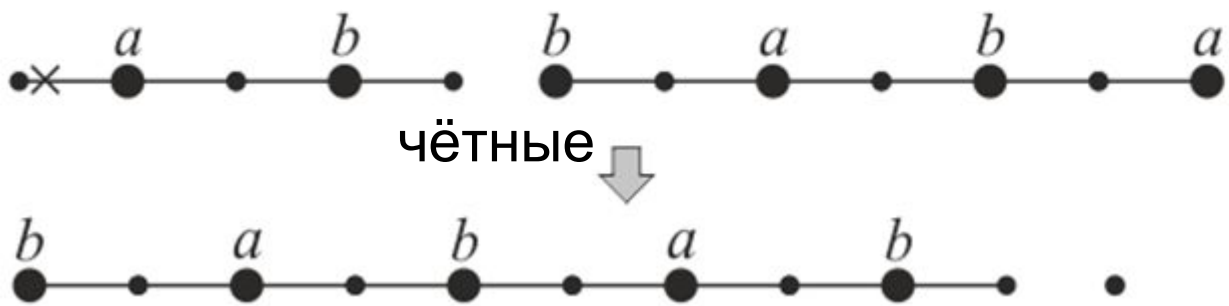
$1a+1b = 1_b{}^*$ нечёт чётная

$3a+2b=1_a{}^*$



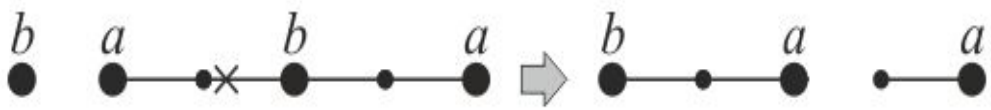
$3a'+2b'= 1_a{}'$

$3^*+2 = 1_b{}^*$



$3a+\underline{3b}=3^*$ и $\underline{2a}+2b=\underline{2}+1'_a$

a) 

b) 

Качество комбинации (=цепочки) операций ***o*** по определению равно ***P*(*o*, {все *типы в* <u>*Ar*</u>}) =**

<u>***A*(*Ar*) – *A*(*o*(*Ar*)) – *c*(*o*)**</u>,

где ***Ar*** есть 2 или 3 или 4 цепи из графа ***G***, к которым применяется ***o***. Таких выделенных комбинаций операций ***o*** будет 51 штука.

Качества семи операции SM на конкретных типах указаны в квадратных скобках, **проверьте!** : $1a+1b=1^*_b$ $[w_a+w_b]$,

$3a+2b=1^*_a$ $[w_b]$,

$3a+2b'=1^*_a$ $[w_a]$,

$3a'+2b^*=1^*_a$ $[w_a]$,

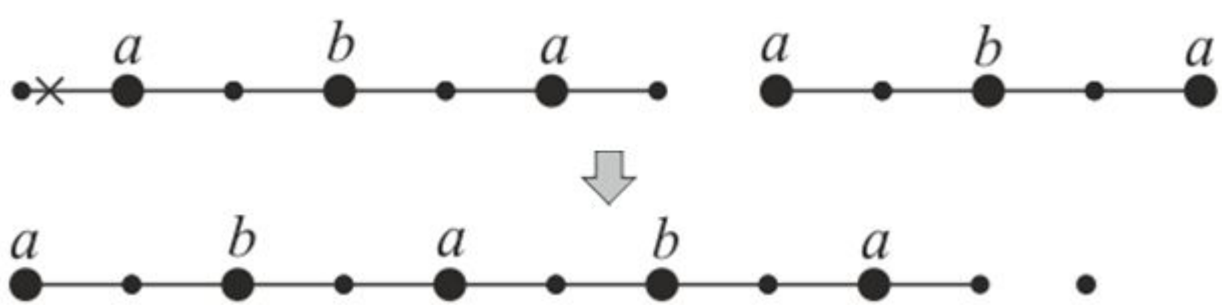
$3a'+2b'=1'_a$ $[w_a]$,

$\underline{3b}+\underline{2a}=\underline{1}_{\cancel{b}}$ $[w_b]$,

$3^*+\underline{\underline{2}}=1^*_b$ $[w_a+w_b-1]$.

В качестве примера вычислим **качество SM**-операции **на** $1a+1b=1^*_b$: слева нечётные цепи, справа чётная. И $s=\{o\}$, $c(s)=1$. Тогда: ***d***, ***c***, ***S*** не меняются, ***B*** и $\boldsymbol{K_b}$ уменьшается на 1, ***f*** и ***D*** уменьшаются на 2. По Лемме 1 получим w_a+w_b.

В качестве примера вычислим **качество SM**-операции **на** $3a+2b=1^*_a$: слева нечётные цепи, справа чётная. И $s=\{o\}$, $c(s)=1$. Тогда: ***d***, ***c***, ***S*** не меняются, ***B***, $\boldsymbol{K_b}$ и ***D*** уменьшается на 1, ***f*** уменьшаются на 2. По Лемме 1 получим w_b.



Взаимодействиями являются **семь** выше указанных **S*M***-операций на указанных **политипах = множествах типов аргументов в одном из взаимодействий**. **Политип взаимно однозначно соответствует взаимодействию !**

И ещё взаим-ия: $1a+2=2a$ $[w_a+w_b-1]$, $1b+2=2b$ $[w_a+w_b-1]$, $3^*+1a=3a$ $[w_a+w_b-1]$, $3^*+1b=3b$ $[w_a+w_b-1]$, $1a+2b=2^*$ $[w_b]$, $1a+2b'=2$ $[w_a]$, $1b+2a=2$ $[w_b]$, $3a+1b=3^*$ $[w_b]$, $3a'+1b=3^*$ $[w_a]$, $3b+1a=3^*$ $[w_b]$; $3a+3b=3^*$ $[w_b-w_a]$, $2b+2a=2+1'_a$ $[w_b-w_a]$.

И ещё взаимодействия: **ОМ**-операция на **политипах**

$1a+1a=3a$ $[w_a+w_b-1]$, $1b+1b=3b$ $[w_a+w_b-1]$.

В политипе типы могут повторяться.

Политип – множество, а не последовательность !

И ещё 3-арные взаимодействия с **политипами**: $3^*+(1a+2b)=1^*_b$ [w_a+2w_b-1], $3^*+(1a+2b')=1^*_b$ [$2w_a+w_b-1$], $3^*+(1b+\underline{2a})=1^*_b$ [w_a+2w_b-1], $(3a+1b)+\underline{2}=1^*_b$ [w_a+2w_b-1], $(3a'+1b)+\underline{2}=1^*_b$ [$2w_a+w_b-1$], $(\underline{3b}+1a)+\underline{2}=1^*_b$ [w_a+2w_b-1], $1a+(1a+2b)=2a$ [w_a+2w_b-1], $1a+(1a+2b')=2a$ [$2w_a+w_b-1$], $1b+(1b+\underline{2a})=2b$ [w_a+2w_b-1], $(\underline{3b}+1a)+1a=3a$ [w_a+2w_b-1], $(3a+1b)+1b=3b$ [w_a+2w_b-1], $(3a'+1b)+1b=3b$ [$2w_a+w_b-1$], $(3a+\underline{2})+\underline{2}=2a^*$ [w_a+2w_b-2], $(3a'+\underline{2})+\underline{2}=2a$ [$2w_a+w_b-2$], $(\underline{3b}+\underline{2})+\underline{2}=2b$ [w_a+2w_b-2], $3^*+(3^*+\underline{2a})=3a$ [w_a+2w_b-2], $3^*+(3^*+2b)=3b$ [w_a+2w_b-2], $3^*+(3^*+2b')=3b$ [$2w_a+w_b-2$], $(3^*+2b)+\underline{2a}=2^*$ [$2w_b-1$], $(3^*+2b')+\underline{2a}=2^*$ [w_a+w_b-1], $3a+(\underline{3b}+\underline{2})=3^*$ [$2w_b-1$], $3a'+(\underline{3b}+\underline{2})=3^*$ [w_a+w_b-1].

В квадратных скобках качества взаимодействий.

И ещё 4-арные взаимодействия с политипами:

$(\underline{3b}+1a)+(1a+2b)=1^*_b$ [w_a+3w_b-1], $(\underline{3b}+1a)+(1a+2b')=1^*_b$ [$2w_a+2w_b-1$], $(3a+1b)+(1b+\underline{2a}) =1^*_b$ [w_a+3w_b-1], $(3a'+1b)+(1b+\underline{2a})=1^*_b$ [$2w_a+2w_b-1$]; $3^*+((3^*+2b^*)+\underline{2a})=1^*_b$ [w_a+3w_b-2], $3^*+((3^*+2b')+\underline{2a})=1^*_b$ [$2w_a+2w_b-2$], $(3a+(\underline{3b}+\underline{2}))+\underline{2}=1^*_b$ [w_a+3w_b-2], $(3a'+(\underline{3b}+\underline{2}))+\underline{2}=1^*_b$ [$2w_a+2w_b-2$].

Наши операции имеют аргументы: DM – какие рёбра и кого с кем соединить ребром, SM – какое ребро и с кем соединить ребром, OM – какие вершины соединить ребром, Cut – какое ребро удалить, Rem – какую вершину удалить !

Итак, цепь определяется её типом и размером! Имеется 17 типов цепей (конечное число), а самих цепей бесконечное число. **Политипов столько, сколько взаимодействий !**

Обозначим G^c множество цепей в *G*.

Именно, над цепями выполняются: операции или их композиции!, т.е. такая композиция ***f*** операций переводит

$$f : G^c \;\rightarrow\; G^c$$

Пусть ***f***(***x***,***y***,***z***) взаимодействие на цепях с политипом {***t1***, ***t2***, ***t3***}.

Аргументов может быть 2 цепи или 3 цепи или 4 цепи.

Элемент □ – множество цепей из G^c , соответствующее какому-то одному политипу, т.е. определённому взаимодейст-вию ***f***. Т.е. □ – множество, к которому можно применить ровно одно взаимодействие ***f***.

Область X в ***G'*** – множество элементов в ***G'*** попарно не пересекающихся как множества;

т.е. пусть □,□ □ ***X***, тогда цепи □ и □ не пересекаются.

Но типы элементов □ и □ могут даже совпадать !!

Максимальная область M – это область, на которой достигает максимума функция (от области ***X***, «качество области ***X***») равная

$$P(X) = \sum_{\alpha \in X} P(\alpha)$$

, где $P(\alpha)$ качество политипа □, т.е. качество соответствующе-го взаимодействия ***f***.

<u>Одинаково</u>: писать политип □ или взаимодействие ***f*** .

Обозначим $x_{□}$ или x_f число элементов в искомой области ***M*** с данным политипом □. Эти x_f элементов автоматически **не пересекаются**; x_f □**0**.

Как найти такую область ***M*** ?

ИТАК, весь алгоритм M **определяется:**

(следующие шаги называются **этапами**):

0) по данным структурам ***a*** и ***b*** образовать общий граф ***G=a+b***;

1) в графе ***G*** <u>вырезать все обычные рёбра</u> (в произвольном порядке); получим граф ***G'*** ;

2) для множества цепей G^c в ***G'*** (точнее, <u>наборов цепей под некоторую композицию</u> ***f***) найдём <u>максимальную область</u> ***M***; и одновременно выполним все композиции из ***M*** (на их аргументах); получим ***G''*** . Волшебный этап алгоритма;

3) к ***G''*** применим автономный алгоритм. □

Как найти максимальную область *M* ?

Для этого используем целочисленное линейное программирование (ЦЛП): каждому взаимодействию ***f*** □ □ сопоставим <u>целочисленную</u> □0 переменную x_f, значение которой <u>должно равняться</u> числу непересекающихся элементов для этого ***f*** в искомой ***M***. (Тогда ***f*** параллельно применим x_f <u>раз к элементам политипа</u> ***f*** .)

<u>Ограничение</u> на вектор $\{x_f\}$: для каждого <u>типа</u> ***t***, представленного в данном ***G'***, и каждого <u>взаимодействия</u> ***f*** :

$$\sum_f c_{tf} \cdot x_f \le l_t$$

где l_t – число всех цепей типа ***t*** в ***G'*** и c_{tf} – число типов ***t*** среди аргументов ***f*** (равное 0, 1 или 2). Положим

$$F(\{x_f\}) = \sum_f P(f) \cdot x_f$$

Максимизируем целевую функцию *F* за линейное время.

Здесь ***P***(***f***) – качество ***f***, т.е. качество его политипа □ .

ЗАВЕРШЕНО описание алгоритма (для одного из соотношений цен) !

Лемма 4 (главная-2). *Для нашего Алгоритма выполняется неравенство треугольника.* □

Из этой Леммы 4 сразу следует точность нашего Алгоритма.

Таким образом, центральный результат нашей работы состоит в том, что существует процедура, которая выдаёт <u>систему участков</u>, для которой верна Лемма 4
(здесь важны как вид $T(G)$, так и система участков, выделяемая нашим Алгоритмом).

Проблема 2. Найти другое пригодное в алгоритме семейство □1.

Определение 7: b) Множество □ называется **полным**, если его нельзя расширить взаимодействием, которое <u>строго увеличивает</u> $P(\boldsymbol{M})$. □

Множество взаимодействий □, приведённое выше, **полное**.

Вектор чисел $\{x_f\}$ в задаче нахождения максимальной области $\boldsymbol{M}$ определяется простым алгоритмом ЦЛП за логарифмическое время.

2-ая задача = задача РЕКОНСТРУКЦИИ (**без паралогов**)

была рассмотрена в начале этого семестра. Она состоит

в алгоритмическом продолжении (=реконструкции)

структур (=множеств цепей и циклов)

с **листьев дерева на его внутренние вершины**.

Продолжение этих исследований состоит в рассмотрении и

Задач преобразования структур и реконструкции структур

с <u>повторением в них имён</u>

(= рассмотрение структур с **паралогами**),

когда <u>цены всех операций равны или не равны</u>.

Следствие 1 к Лемме 1. Эти операции ***f*** и их качества ***P*(*f*)** поднимаются на фактор-множество по типам цепей, т.е. фактически являются <u>операциями на типах цепей</u>.

$$\forall x, y, x', y' (x \sim x', y \sim y' \Rightarrow f(x, y) \sim f(x', y'))$$

Определение 6. ***ПОЛИТИПОМ*** называется последовательнос-ть типов $t_1, \ldots, t_n$ (например, <1a,1b>, что лучше записывать 1a+1b или $1a+1b = 1_b^*$).

ВЗАИМОДЕЙСТВИЕМ* с политипом $t_1, \ldots, t_n$ называется композиция операций *f (часто это – то-лько одна операция) с **аргументами-<u>цепями типов</u> $t_1, \ldots, t_n$, которая вместе с качеством поднимается на фактор-множество и выполняется $P(f(t_1, \ldots, t_n)) > 0$**

Доказательство Следствий.

1) При замене цепи в том же типе тип значения не меняется. Перебором.

2) Цепи-аргументы в s обозначим G. При замене цепи в том же типе в G в разности $A(G) – A(s(G))$ число нечёт цепей f не меняется (2 и 0); S равно 0; D не меняется;

d в 1-м и 2-м слагаемом не меняется и сокращается;

B во 2-м слагаемом становится равным B-1 (здесь B из 1-го слагаемого) и при вычитании приносит 1, равную цене SM-операции. □

Следствие 1 к Лемме 1. Эти операции ***f*** и их качества ***P*(*f*)** поднимаются на фактор-множество по типам цепей, т.е. фактически являются операциями на типах цепей.

$$\forall x, y, x', y' (x \sim x', y \sim y' \Rightarrow f(x,y) \sim f(x',y'))$$

Порядок цепей в элементе однозначно определяется политипом, поэтому множество цепей для любого политипа неупорядочено!

Качество ***P***(***X***) не меняется при замене цепей в ***X*** с сохранением типа и без нарушения инъективности!