

Министерство образования Республики Беларусь

Учреждение образования  
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет Информационных технологий и управления  
Кафедра Интеллектуальных информационных технологий

**РАСЧЁТНАЯ РАБОТА**  
по дисциплине «Представление и обработка информации в  
интеллектуальных системах»  
на тему  
**Задача нахождения графа конденсации для  
ориентированного графа**

Выполнил:

Б. В. Бурак

Студент группы  
321702

Проверил:

Н. В. Малиновская

Минск 2024

# 1 ВВЕДЕНИЕ

**Цель:** Получить навыки формализации и обработки информации с использованием семантических сетей

**Задача:** Найти граф конденсации для данного ориентированного графа.

## 2 СПИСОК ПОНЯТИЙ

1. **Граф** (рис. 2.1) (абсолютное понятие) - структура, состоящая из следующих объектов:
  - a. Вершины (относительное понятие, ролевое отношение);
  - b. Рёбра (относительное понятие, ролевое отношение) - связи между вершинами.

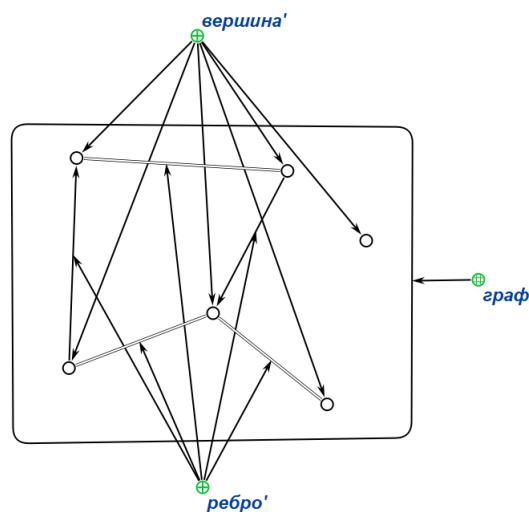


Рисунок 2.1 – Граф

2. **Ориентированный граф**, или орграф (рис. 2.2) (абсолютное понятие) - структура, состоящая из следующих объектов:
  - a. Вершины (относительное понятие, ролевое отношение);
  - b. Дуги (относительное понятие, ролевое отношение) - рёбра, имеющие направление.

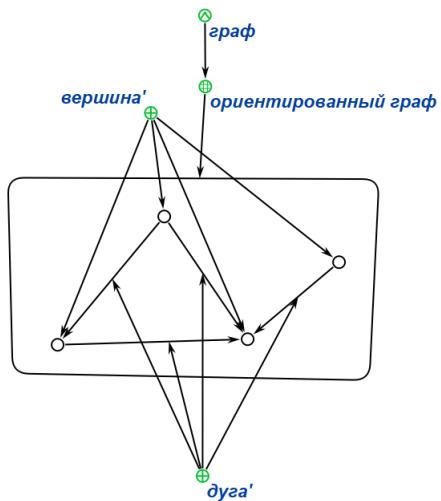


Рисунок 2.2 – Орграф

3. **Транспонированный граф** (рис. 2.3) (относительное понятие, неролевое отношение) - орграф, имеющий по сравнению с изначальным тот же набор вершин и ребёр, но каждое ребро которого имеет противоположное направление.

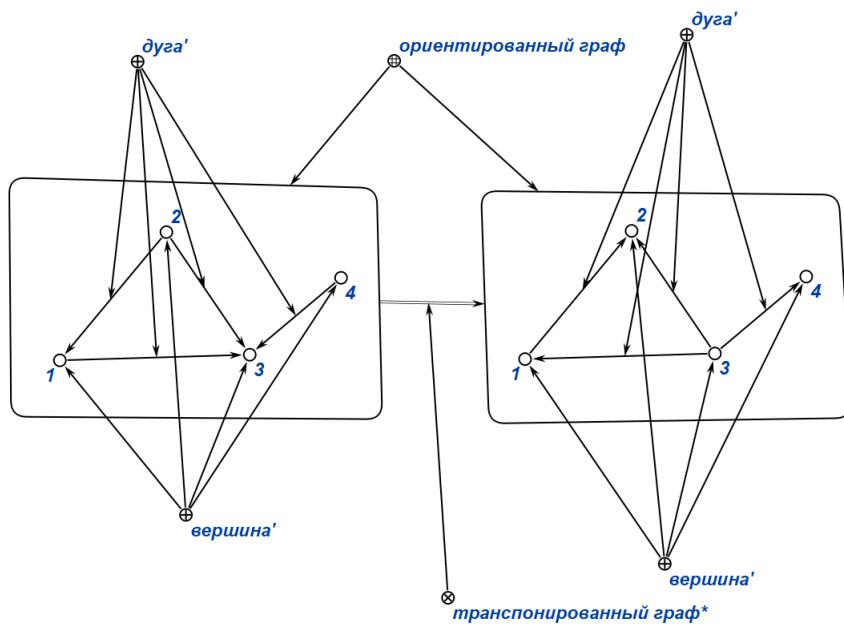


Рисунок 2.3 – Транспонированный граф

4. **Сильная связность графа** (рис. 2.4) (относительное понятие, неролевое отношение) - свойство орграфа, при наличии которого для любых вершин  $u$  и  $v$ , принадлежащих графу, существует путь из  $u$  в  $v$  и из  $v$  в  $u$ .

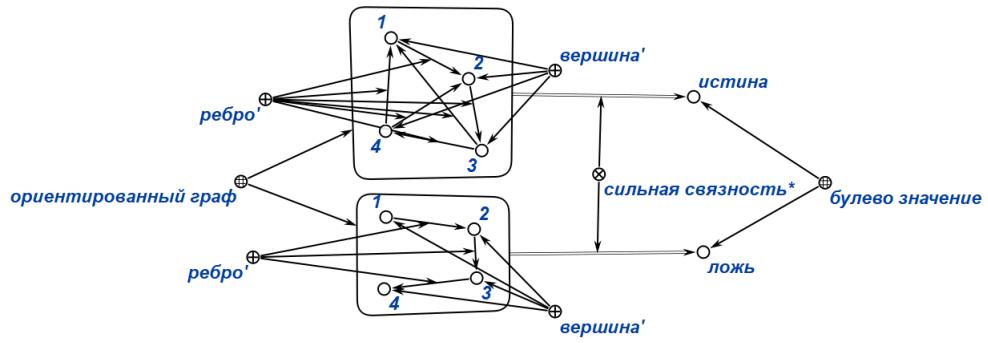


Рисунок 2.4 – Сильная связность графа

5. **Компоненты сильной связности** графа (рис. 2.5) (относительное понятие, неролевое отношение) - максимальные по включению подграфы данного орграфа, обладающие сильной связностью.

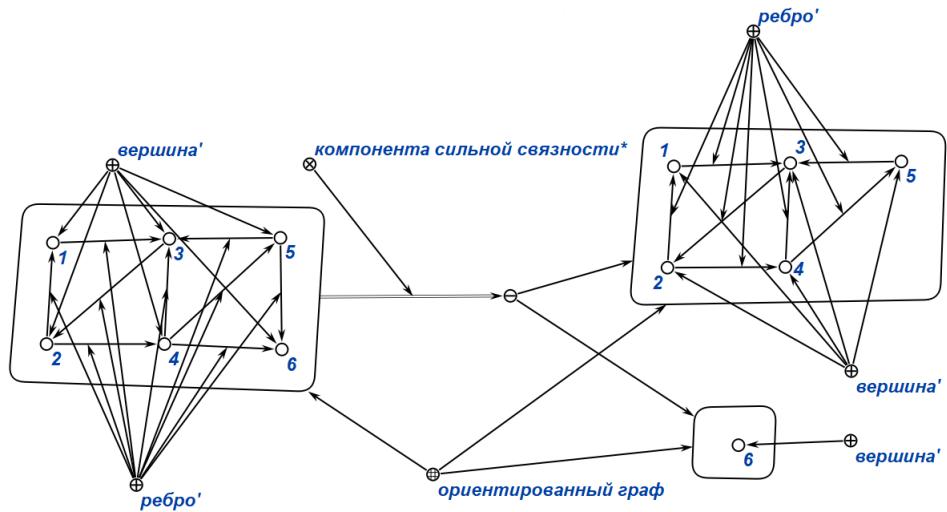


Рисунок 2.5 – Компоненты сильной связности

6. **Граф конденсации** (рис. 2.6) (относительное понятие, неролевое отношение) - граф, в качестве вершин которого выступают компоненты сильной связности исходного орграфа, а рёбра которого показывают наличие рёбер между вершинами исходного графа, принадлежащим разным компонентам сильной связности.

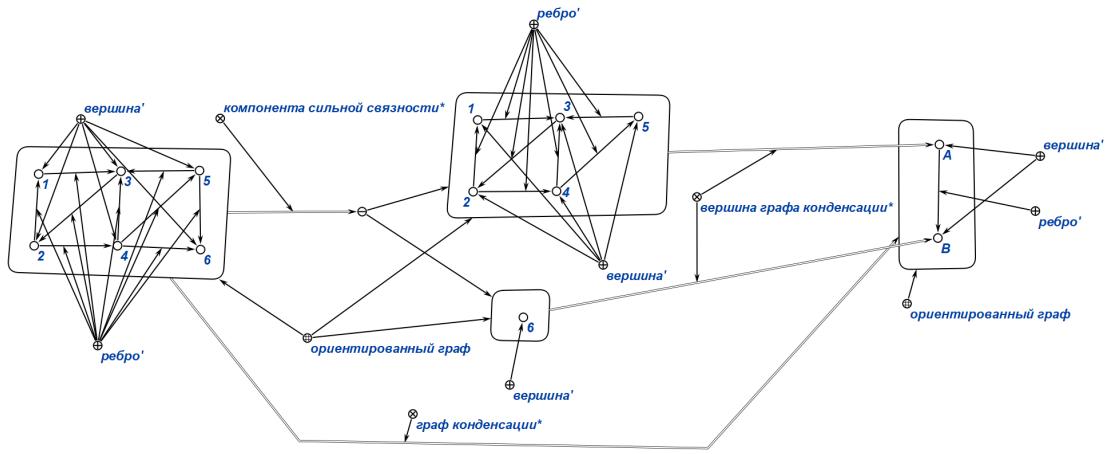


Рисунок 2.6 – Граф конденсации

### 3 ТЕСТОВЫЕ ПРИМЕРЫ

Во всех тестах графы будут приведены в сокращенной форме со скрытыми ролями элементов графа.

#### 3.1 Тест 1

**Вход:**

Найти граф конденсации для данного ориентированного графа.

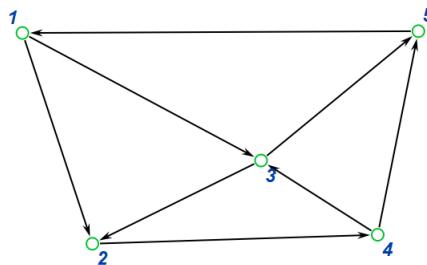


Рисунок 3.1 – Вход теста 1

**Выход:**

Граф конденсации, состоящий из единственной вершины.



Рисунок 3.2 – Выход теста 1

#### 3.2 Тест 2

**Вход:**

Найти граф конденсации для данного ориентированного графа.

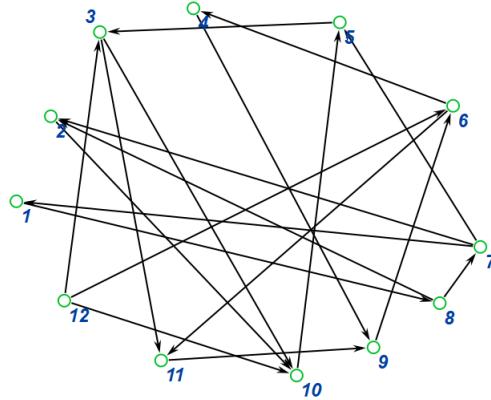


Рисунок 3.3 – Вход теста 2

**Выход:**

Граф конденсации, состоящий из пяти вершин и шести дуг.

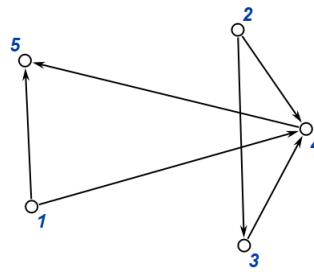


Рисунок 3.4 – Выход теста 2

### 3.3 Тест 3

**Вход:**

Найти граф конденсации для данного ориентированного графа.

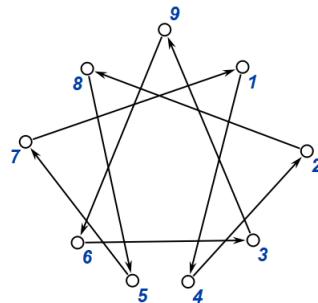


Рисунок 3.5 – Вход теста 3

**Выход:**

Граф конденсации, состоящий из двух вершин без дуги между ними.



Рисунок 3.6 – Выход теста 3

### 3.4 Тест 4

**Вход:**

Найти граф конденсации для данного ориентированного графа.

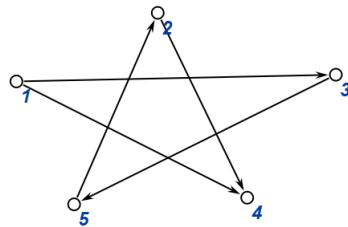


Рисунок 3.7 – Вход теста 4

**Выход:**

Граф конденсации, идентичный исходному графу.

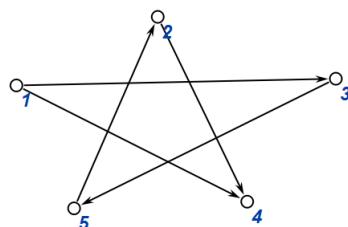


Рисунок 3.8 – Выход теста 4

### 3.5 Тест 5

**Вход:**

Найти граф конденсации для данного ориентированного графа.

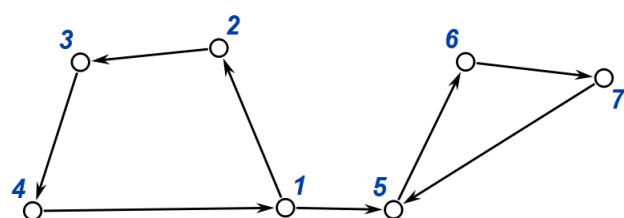


Рисунок 3.9 – Вход теста 5

**Выход:**

Граф конденсации, состоящий из двух вершин и одной дуги.



Рисунок 3.10 – Выход теста 5

## 4 ПРИМЕР РАБОТЫ АЛГОРИТМА В СЕМАНТИЧЕСКОЙ ПАМЯТИ

1. Задание входного графа.

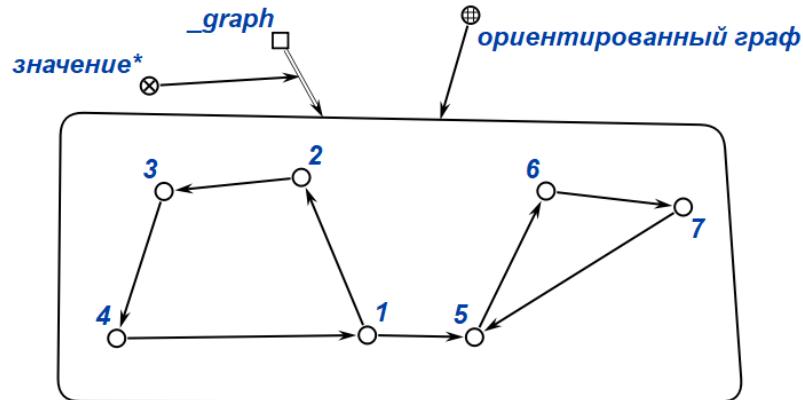


Рисунок 4.1 – Шаг 1

`_graph` получит в качестве значения sc-узел ориентированного графа.

2. Создание множества непосещённых вершин.

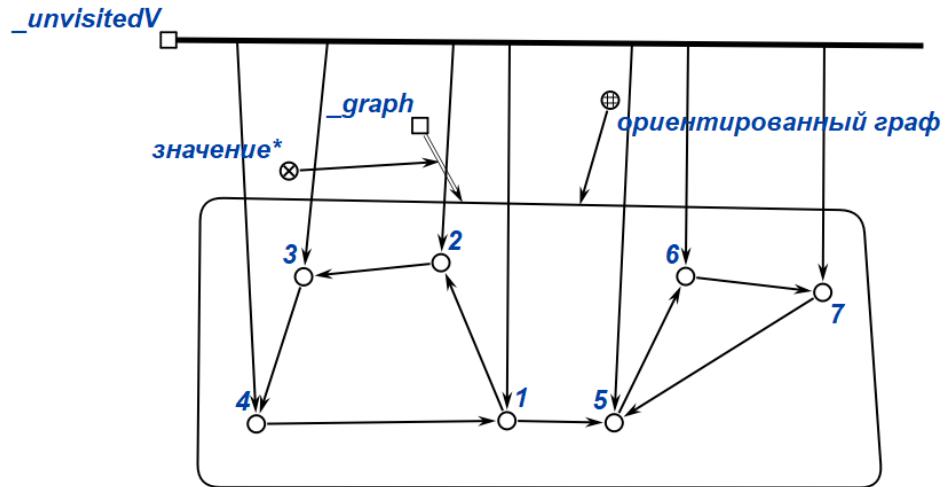


Рисунок 4.2 – Шаг 2

Создаётся множество `_unvisitedV`, в которое входят все вершины `_graph`.

3. Топологическая сортировка графа.

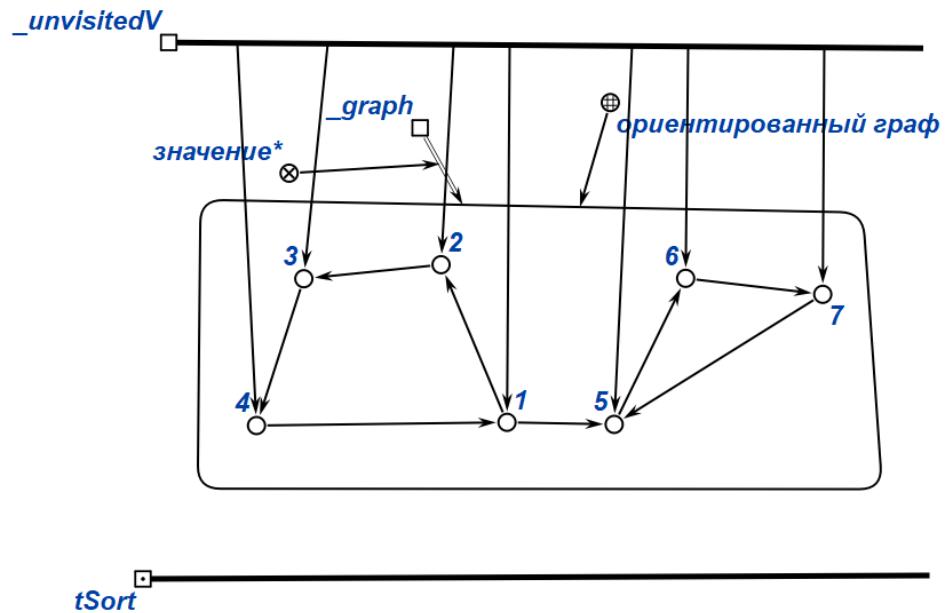
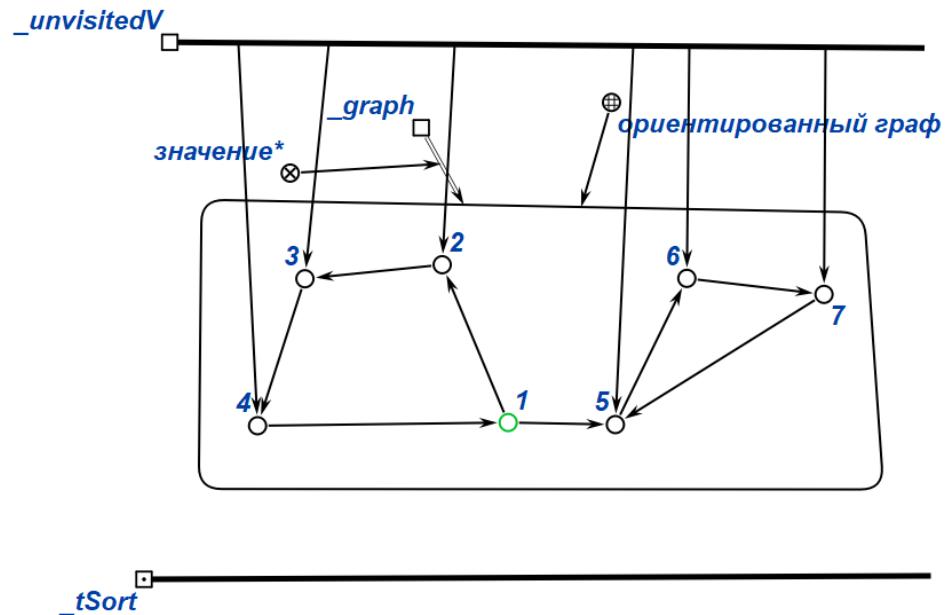
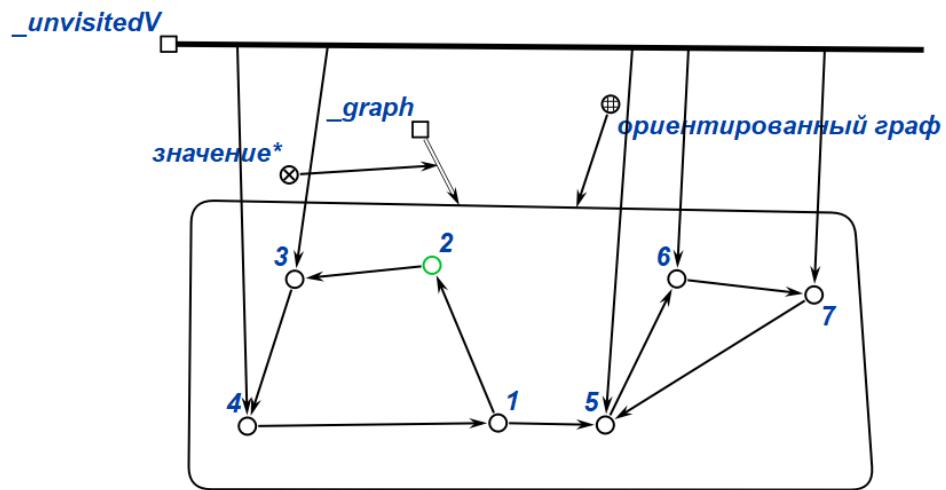


Рисунок 4.3 – Шаг 3

Создаётся пустой список `_tSort`.

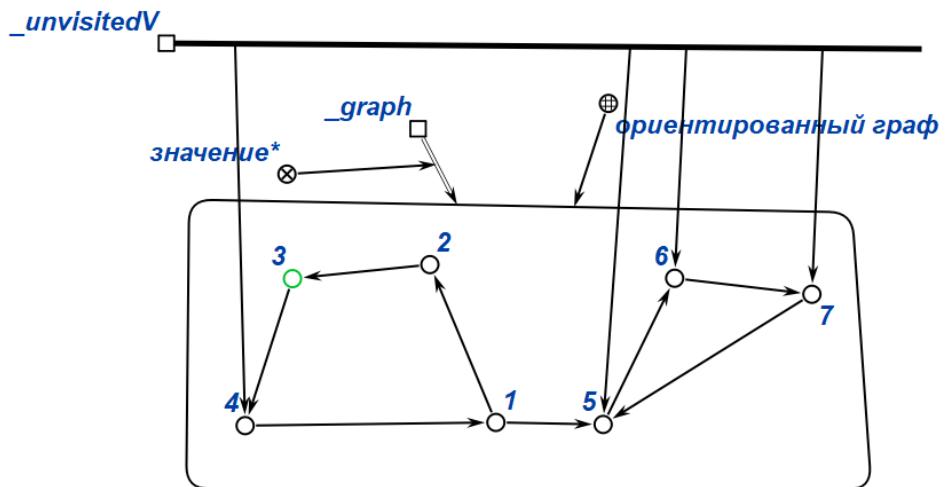


- a. Начинаем обход с вершины 1. Удаляем её из `_unvisitedV`.



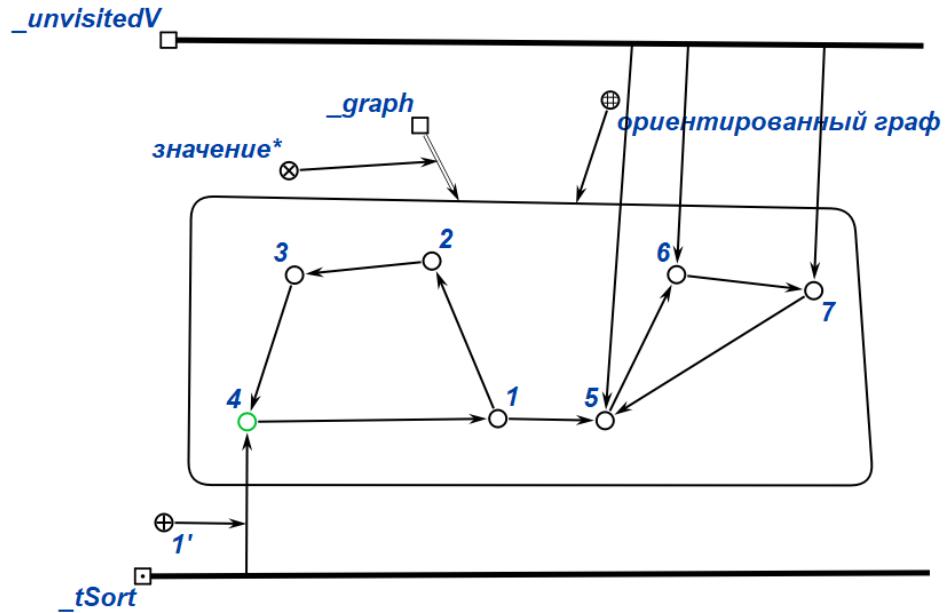
`_tSort`

- b. Переходим на непосещённую вершину с минимальным номером, к которой от вершины 1 есть дуга - 2. Удаляем вершину 2 из `_unvisitedV`.

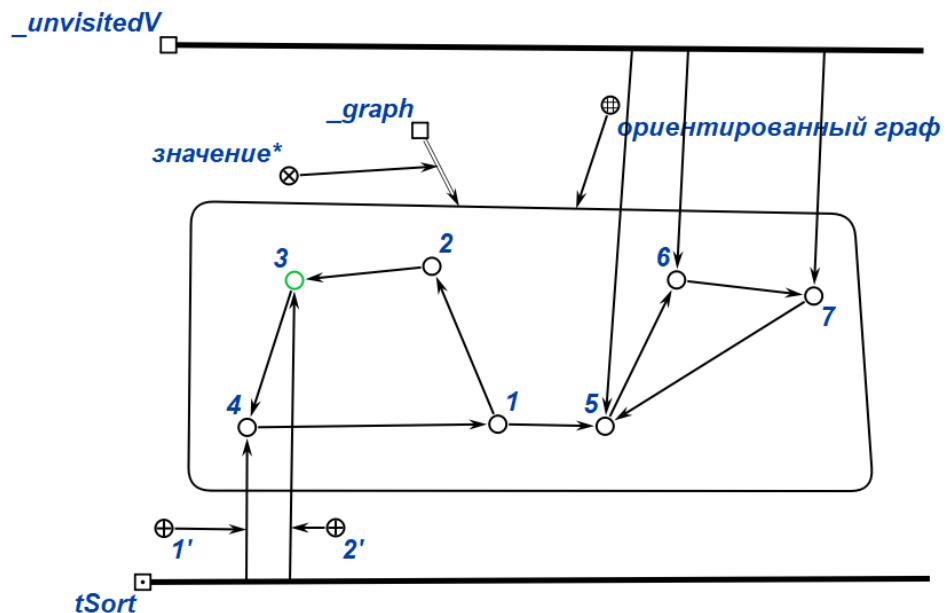


`_tSort`

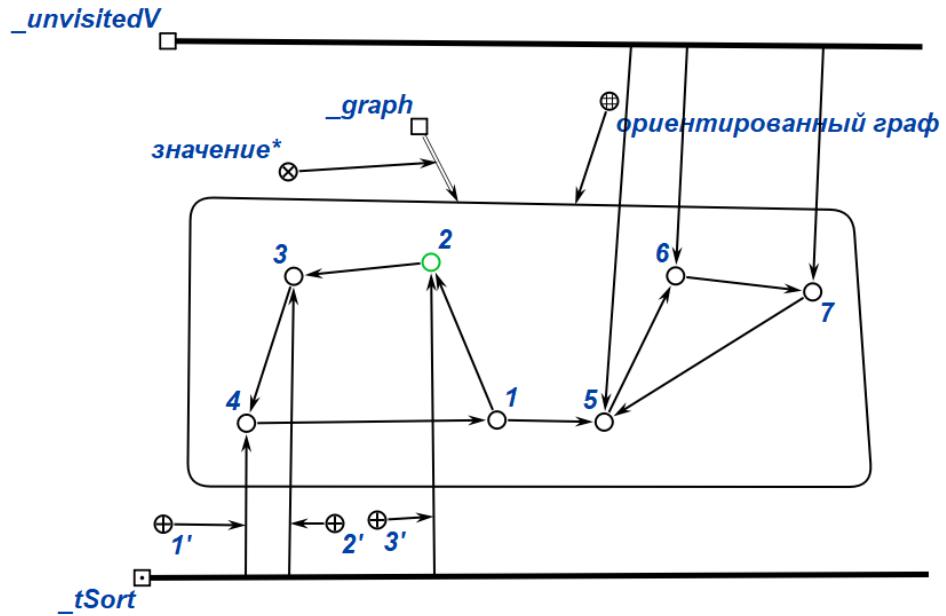
- c. Переходим на единственную непосещённую вершину, к которой от вершины 2 есть дуга - 3. Удаляем вершину 3 из `_unvisitedV`.



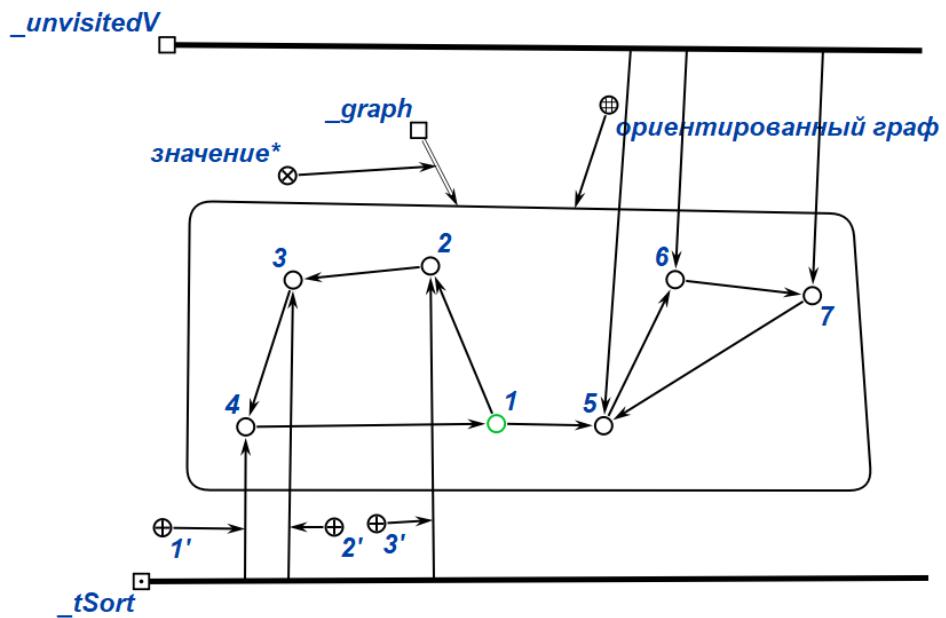
- d. Переходим на единственную непосещённую вершину, к которой от вершины 3 есть дуга - 4. Удаляем вершину 4 из `_unvisitedV`. Т.к от вершины 4 нет дуг к непосещённым вершинам, добавляем её под номером 1 в список `_tSort`.



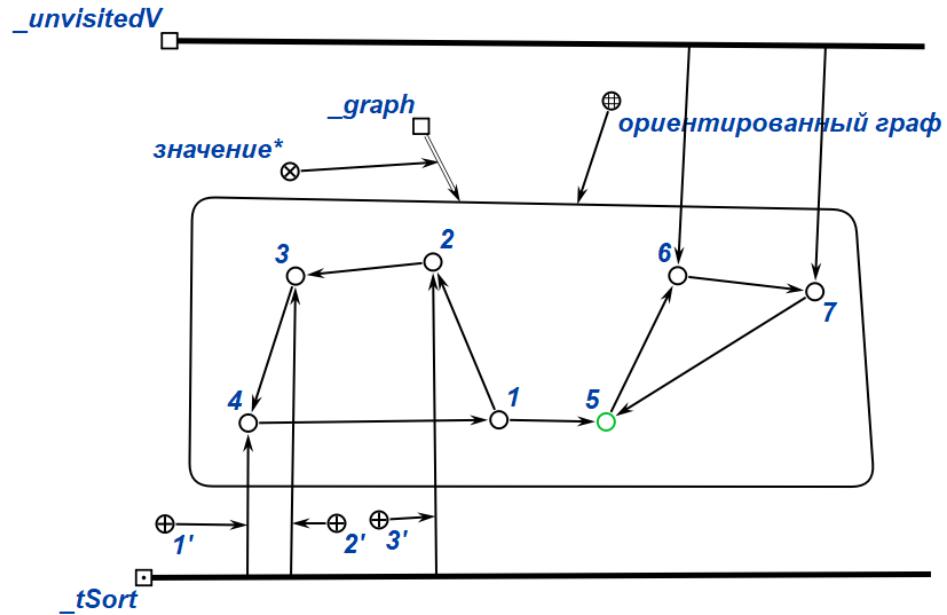
- e. Возвращаемся в вершину 3. Т.к от вершины 3 нет дуг к непосещённым вершинам, добавляем её под номером 2 в список `_tSort`.



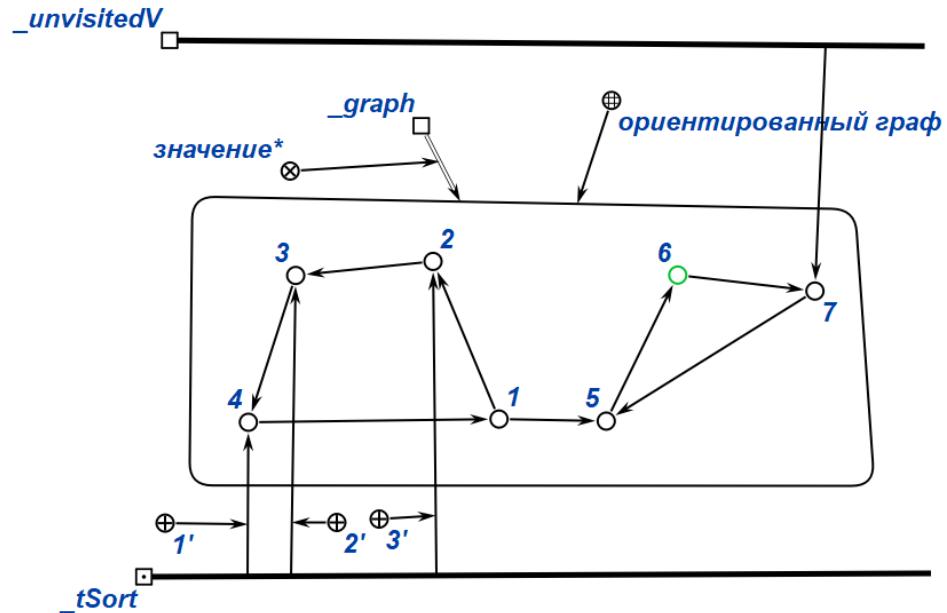
f. Возвращаемся в вершину 2. Т.к от вершины 2 нет дуг к непосещённым вершинам, добавляем её под номером 3 в список `_tSort`.



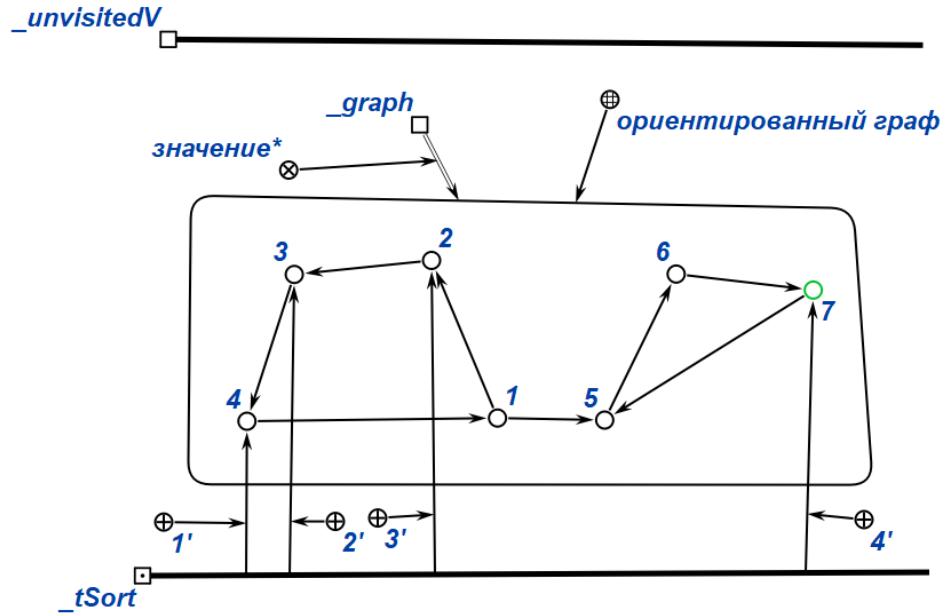
g. Возвращаемся в вершину 1.



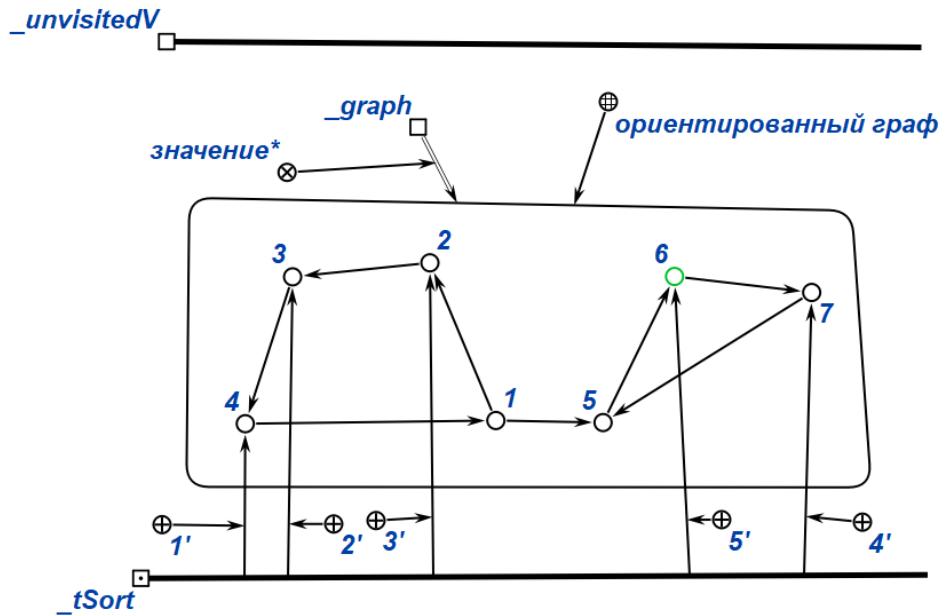
- h. Переходим на единственную непосещённую вершину, к которой от вершины 1 есть дуга - 5. Удаляем вершину 5 из *\_unvisitedV*.



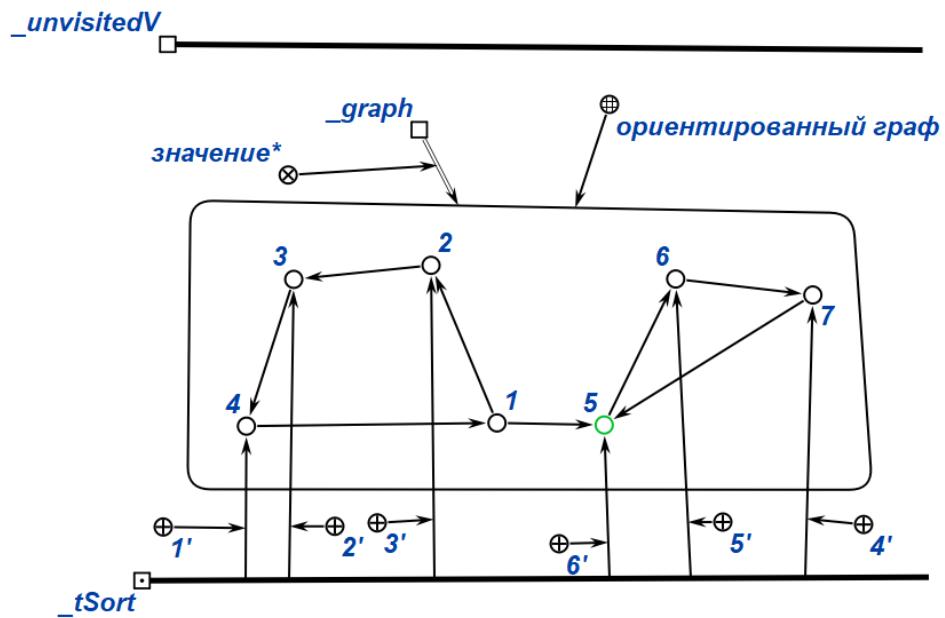
- i. Переходим на единственную непосещённую вершину, к которой от вершины 5 есть дуга - 6. Удаляем вершину 6 из *\_unvisitedV*.



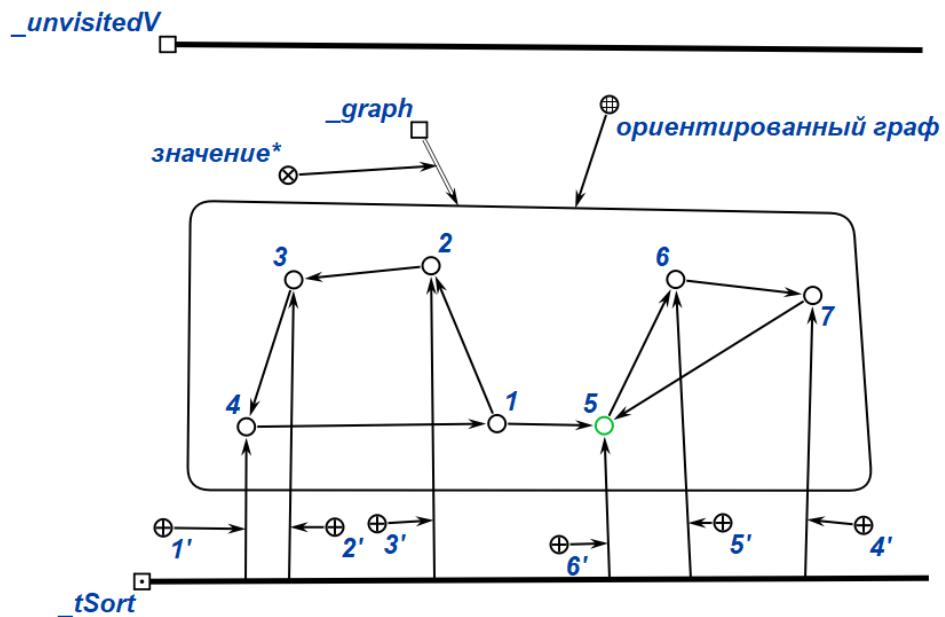
- j. Переходим на единственную непосещённую вершину, к которой от вершины  $6$  есть дуга -  $7$ . Удаляем вершину  $7$  из `_unvisitedV`. Т.к от вершины  $7$  нет дуг к непосещённым вершинам, добавляем её под номером  $4$  в список `_tSort`.



- k. Возвращаемся в вершину  $6$ . Т.к от вершины  $6$  нет дуг к непосещённым вершинам, добавляем её под номером  $5$  в список `_tSort`.



1. Возвращаемся в вершину 5. Т.к от вершины 5 нет дуг к непосещённым вершинам, добавляем её под номером 6 в список `_tSort`.



- m. Возвращаемся в вершину 1. Т.к от вершины 1 нет дуг к непосещённым вершинам, добавляем её под номером 7 в список `_tSort`. Непосещённых вершин не осталось, так что построение списка топологической сортировки графа завершено.

4. Построение транспонированного графа.

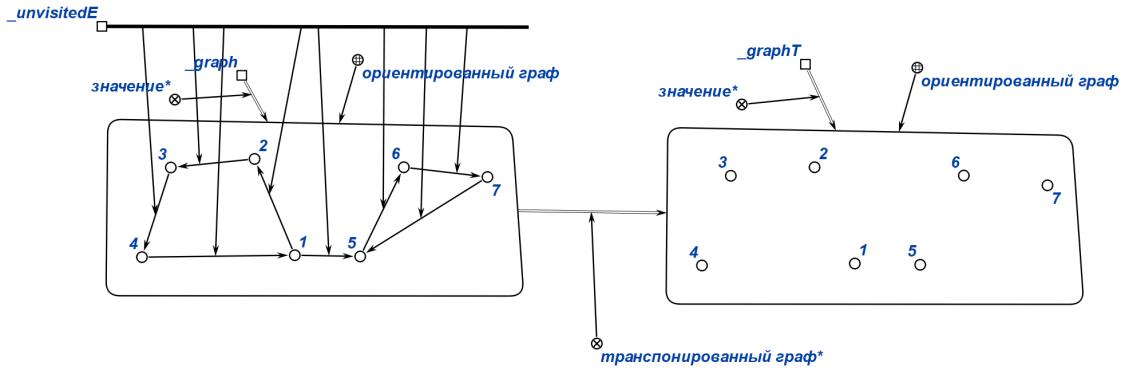
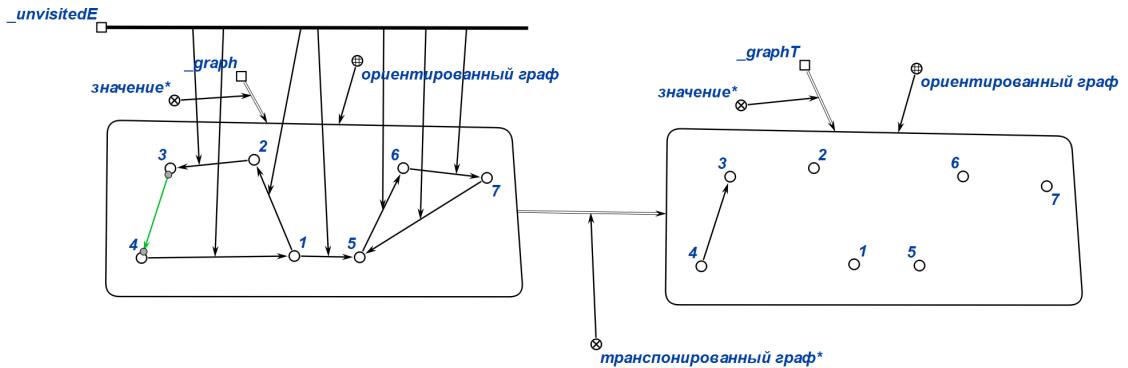
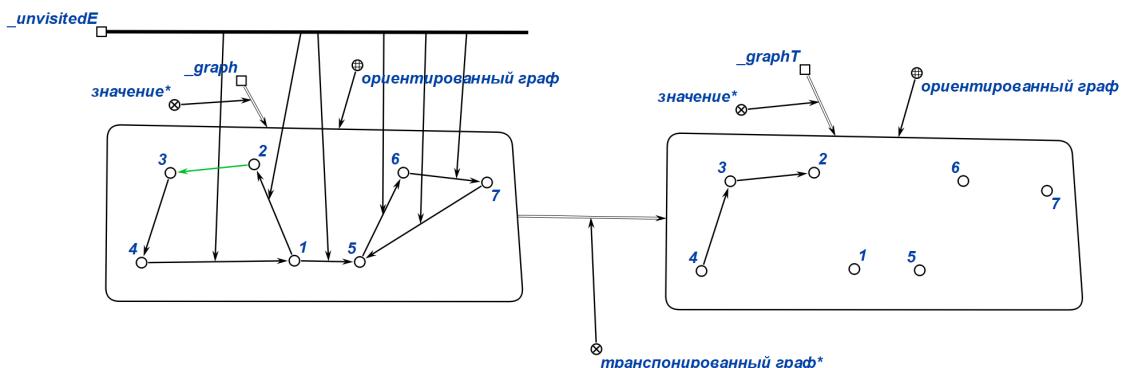


Рисунок 4.4 – Шаг 4

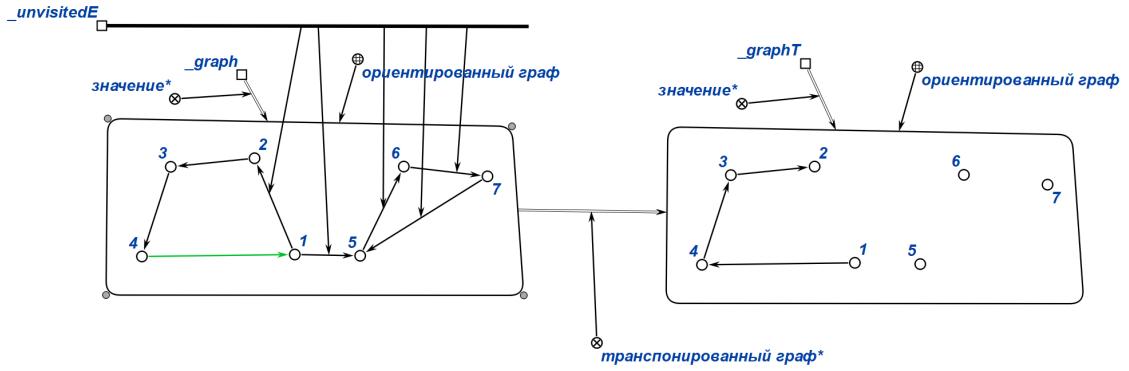
Создаётся граф  $_graphT$ . В него добавляются все вершины графа  $_graphT$ . Создаётся множество  $_unvisitedE$ , в котором содержатся все дуги  $_graph$ .



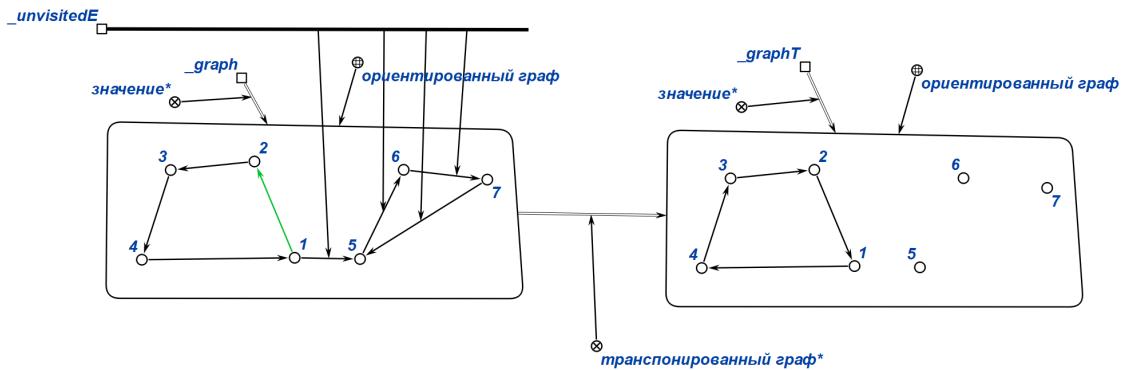
- В  $_graph$  присутствует дуга  $3 \rightarrow 4$ , поэтому записываем дугу  $4 \rightarrow 3$  в  $_graphT$ . Удаляем дугу из  $_unvisitedE$ .



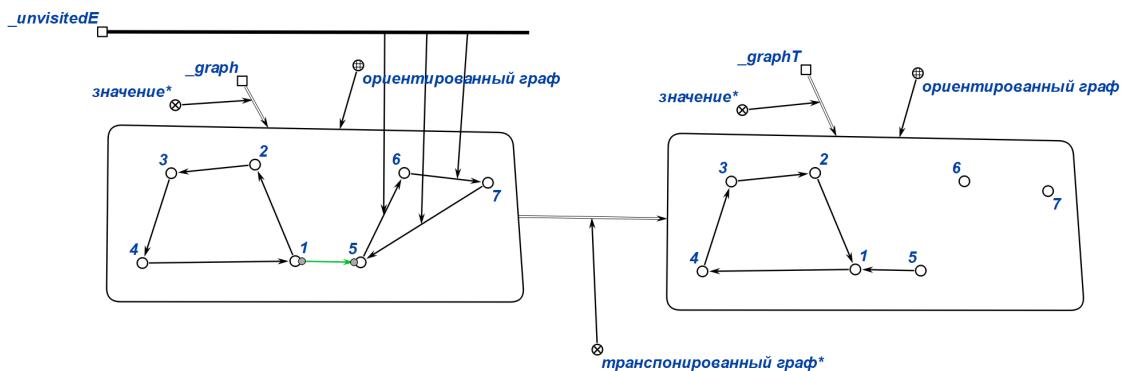
- В  $_graph$  присутствует дуга  $2 \rightarrow 3$ , поэтому записываем дугу  $3 \rightarrow 2$  в  $_graphT$ . Удаляем дугу из  $_unvisitedE$ .



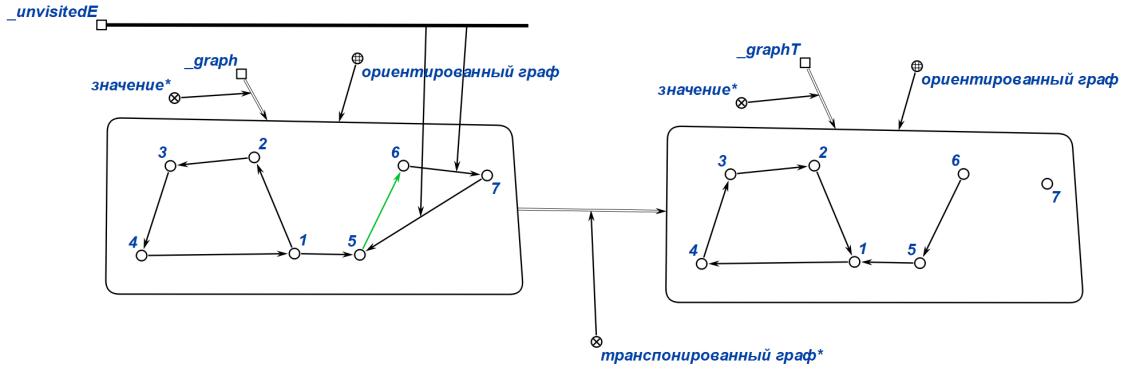
- c. В  $_graph$  присутствует дуга  $4 \rightarrow 1$ , поэтому записываем дугу  $1 \rightarrow 4$  в  $_graphT$ . Удаляем дугу из  $_unvisitedE$ .



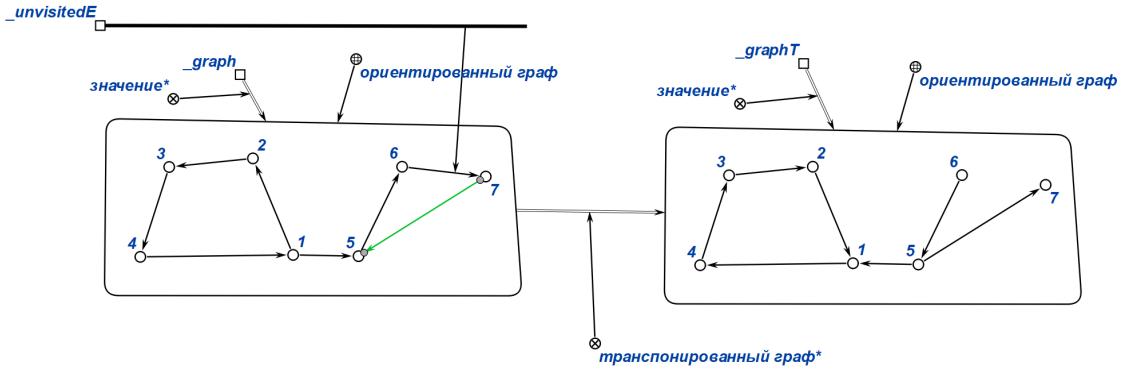
- d. В  $_graph$  присутствует дуга  $1 \rightarrow 2$ , поэтому записываем дугу  $2 \rightarrow 1$  в  $_graphT$ . Удаляем дугу из  $_unvisitedE$ .



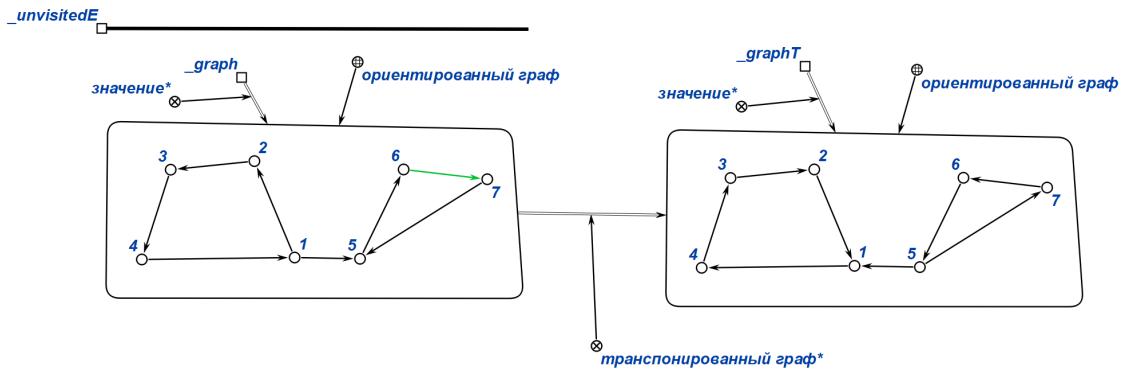
- e. В  $_graph$  присутствует дуга  $1 \rightarrow 5$ , поэтому записываем дугу  $5 \rightarrow 1$  в  $_graphT$ . Удаляем дугу из  $_unvisitedE$ .



f. В  $_graph$  присутствует дуга  $5 \rightarrow 6$ , поэтому записываем дугу  $6 \rightarrow 5$  в  $_graphT$ . Удаляем дугу из  $_unvisitedE$ .



g. В  $_graph$  присутствует дуга  $7 \rightarrow 5$ , поэтому записываем дугу  $5 \rightarrow 7$  в  $_graphT$ . Удаляем дугу из  $_unvisitedE$ .



h. В  $_graph$  присутствует дуга  $6 \rightarrow 7$ , поэтому записываем дугу  $7 \rightarrow 6$  в  $_graphT$ . Удаляем дугу из  $_unvisitedE$ .

Т.к. непосещённых дуг не осталось, построение  $_graphT$  завершено.

5. Построение списка компонент сильной связности.

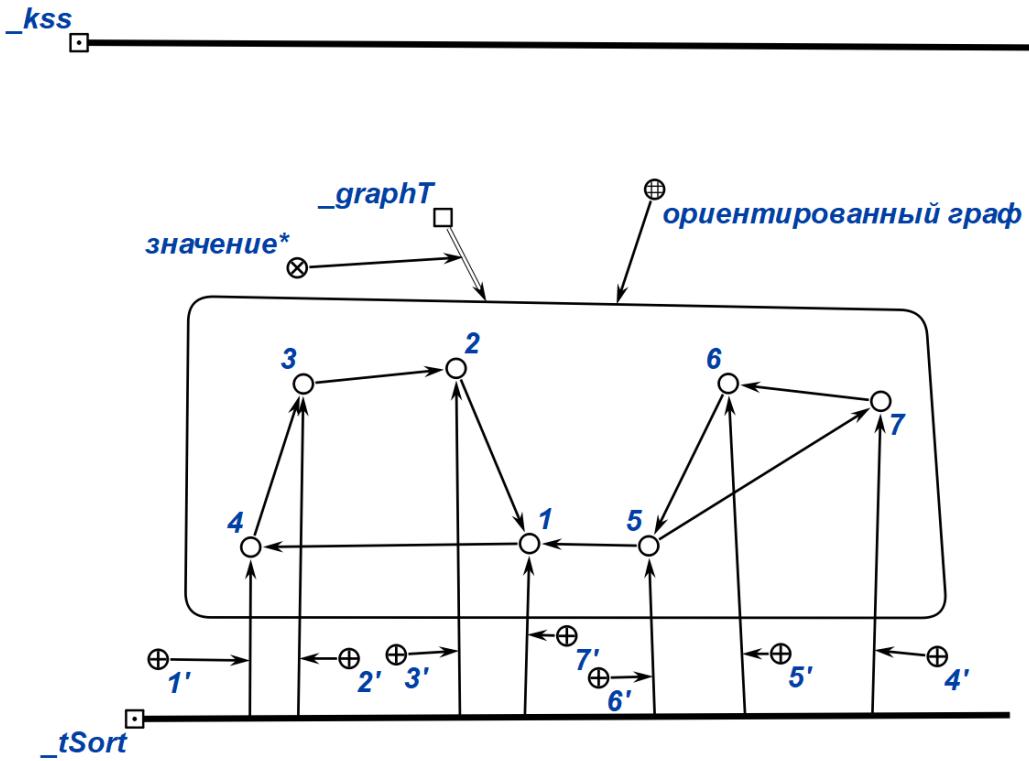
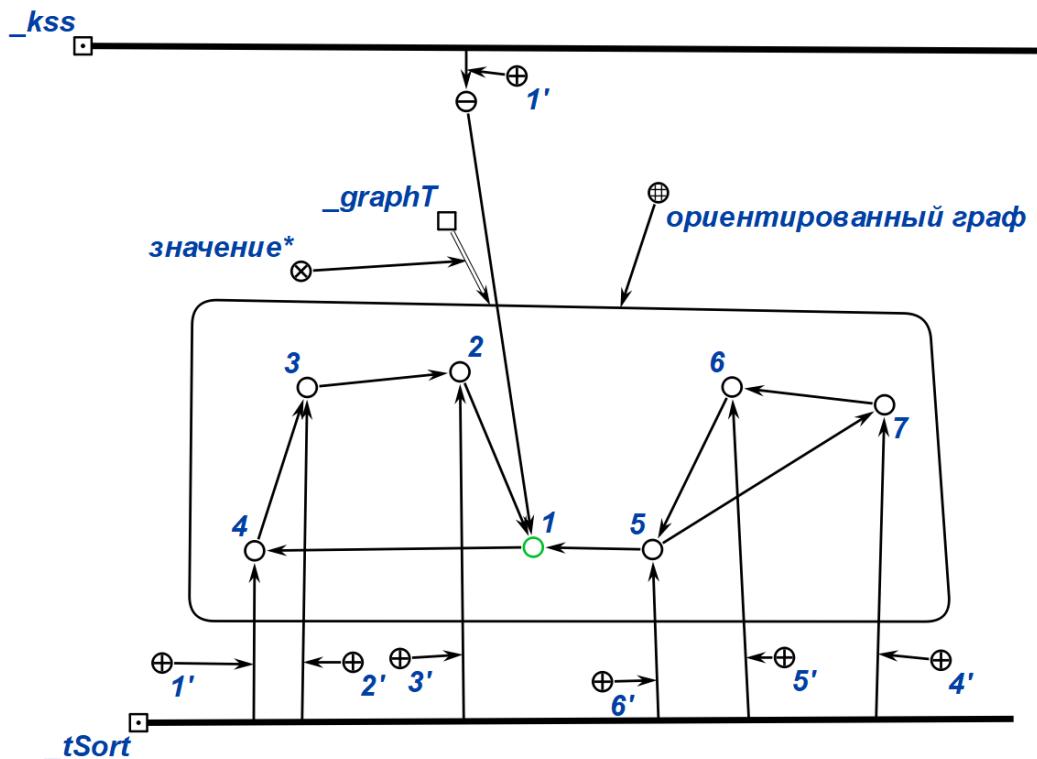


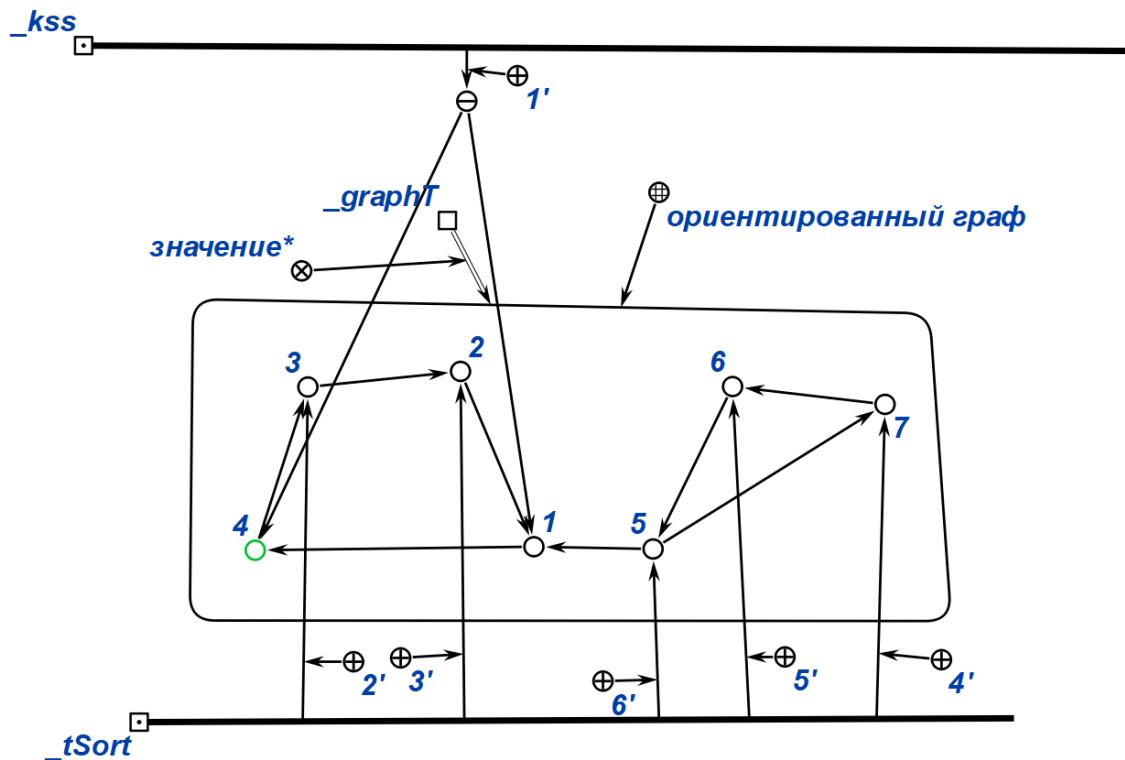
Рисунок 4.5 – Шаг 5

Создаётся пустой список `_kss`.

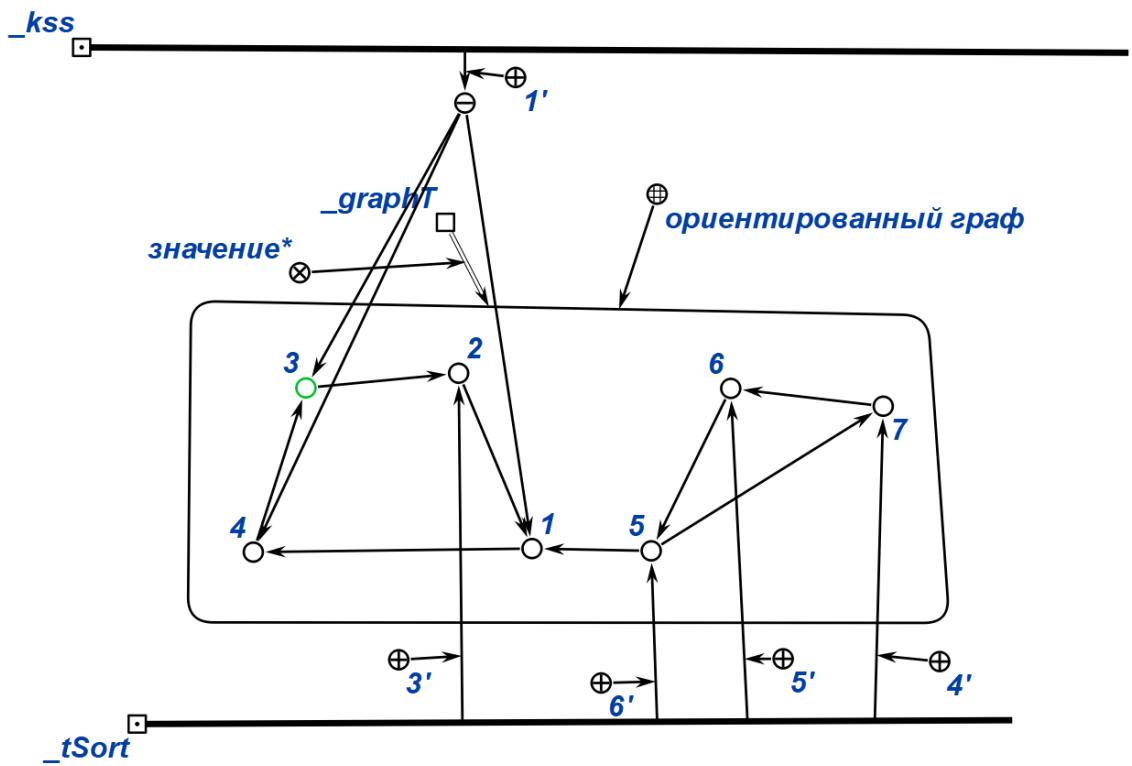


- Начинаем первый обход `_graphT`. В `_kss` добавляется пустое множество, обозначающее вершины графов `_graph` и `_graphT`,

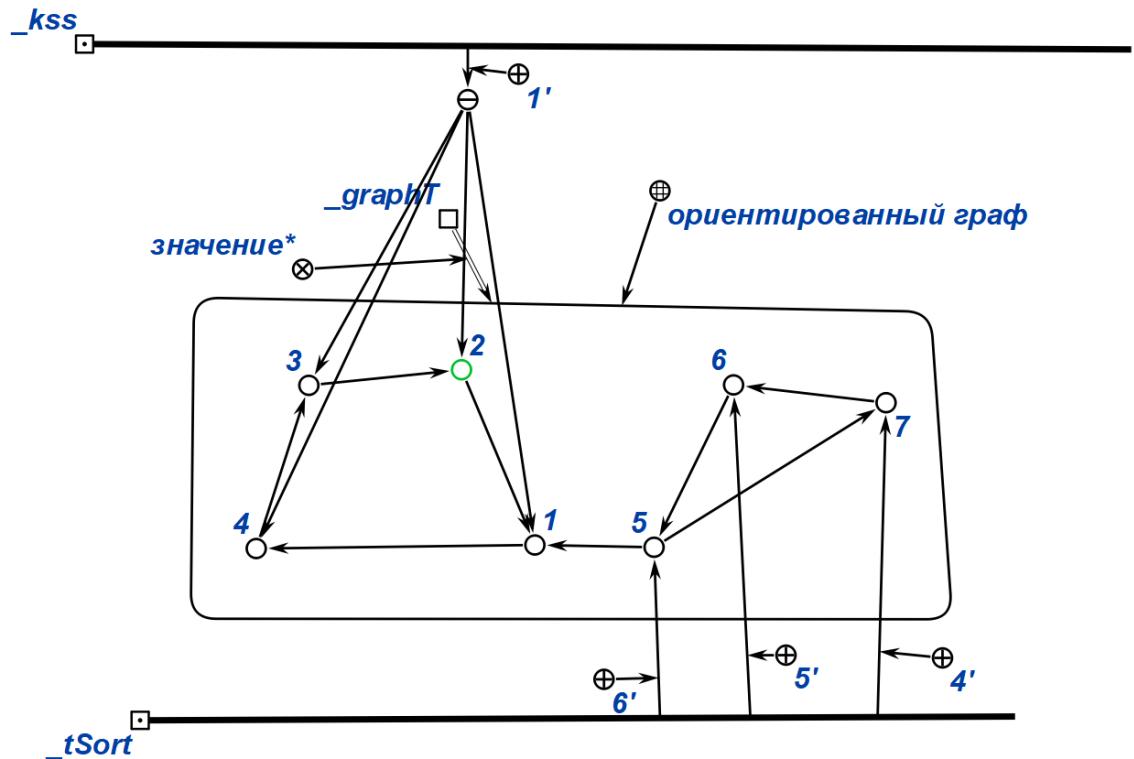
относящиеся к первой компоненте сильной связности. Добавляем в него вершину графа с максимальным номером в `_tSort` - 1 и удаляем её же из `_tSort`.



- Переходим к вершине с максимальным номером в `_tSort`, к которой от вершины 1 есть дуга - 4. Добавляем вершину 4 в множество вершин первой компоненты сильной связности в `_kss` и удаляем её же из `_tSort`.

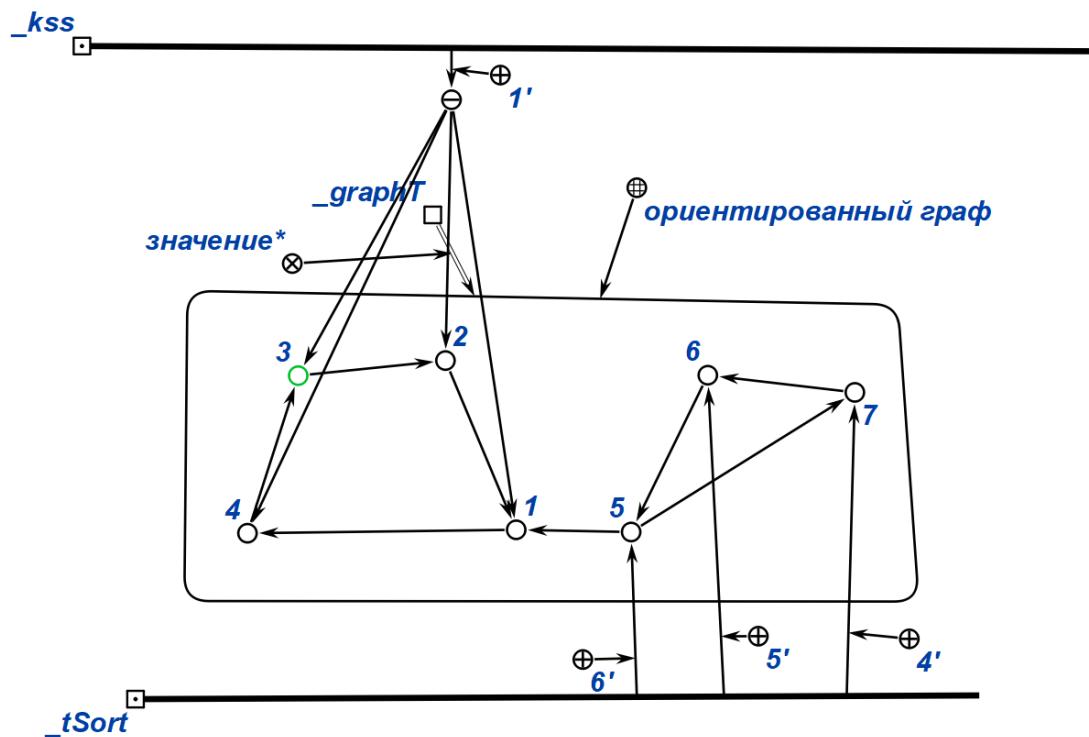


с. Переходим к вершине с максимальным номером в  $_tSort$ , к которой от вершины 4 есть дуга - 3. Добавляем вершину 3 в множество вершин первой компоненты сильной связности в  $_kss$  и удаляем её же из  $_tSort$ .

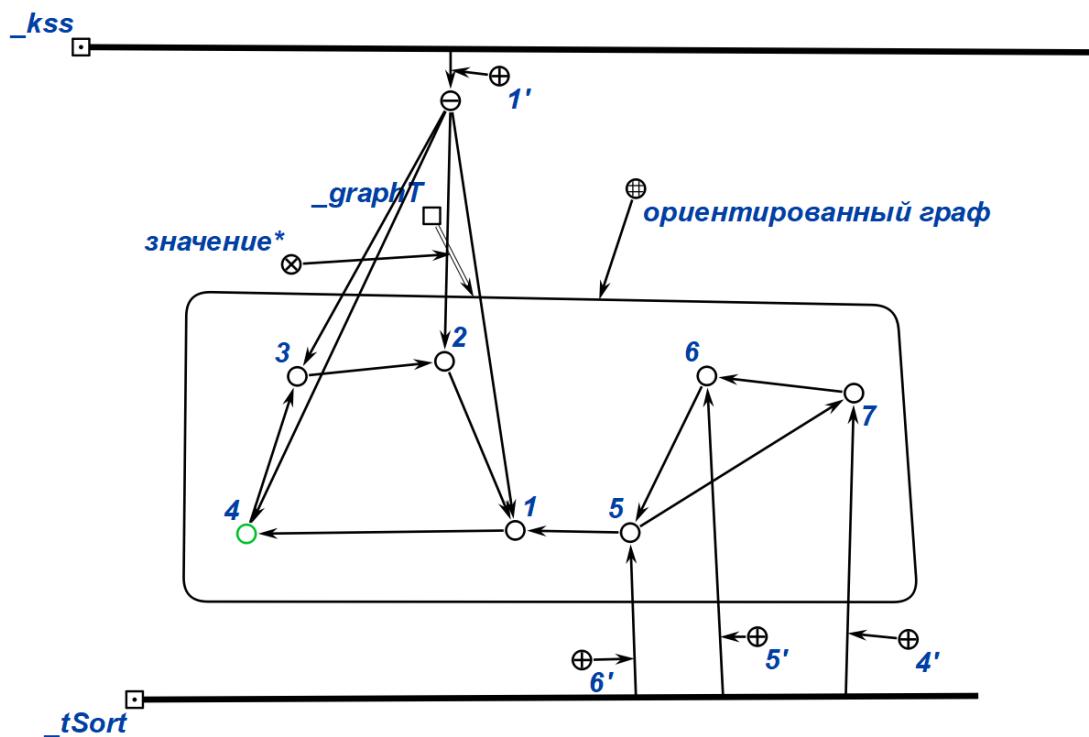


д. Переходим к вершине с максимальным номером в  $_tSort$ , к

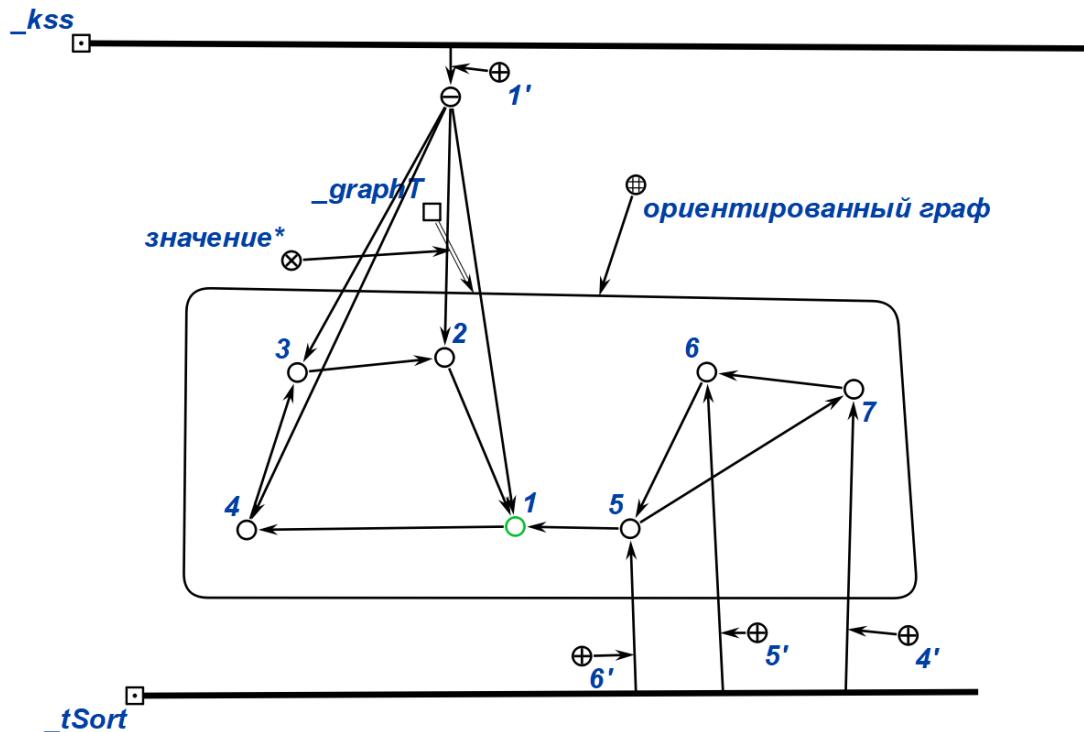
которой от вершины 3 есть дуга - 2. Добавляем вершину 2 в множество вершин первой компоненты сильной связности в `_kss` и удаляем её же из `_tSort`.



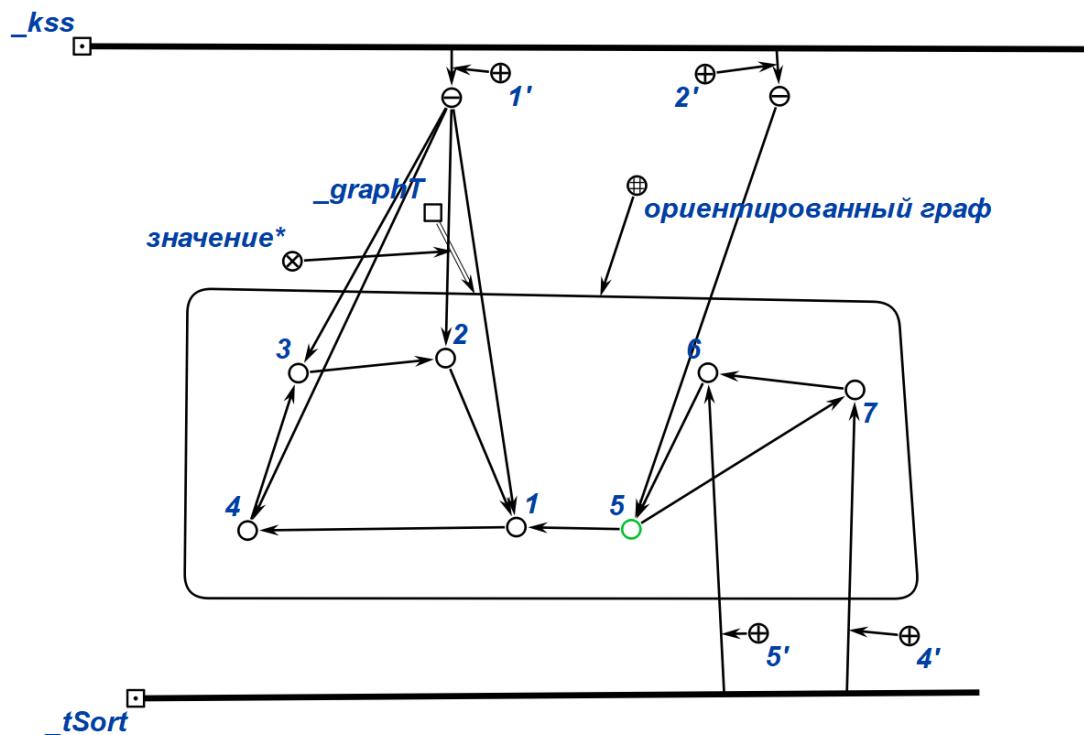
е. Т.к. от вершины 2 нет дуг к вершинам, присутствующим в списке топологической сортировки, возвращаемся к вершине 3.



f. Т.к. от вершины 3 нет дуг к вершинам, присутствующим в списке топологической сортировки, возвращаемся к вершине 4.

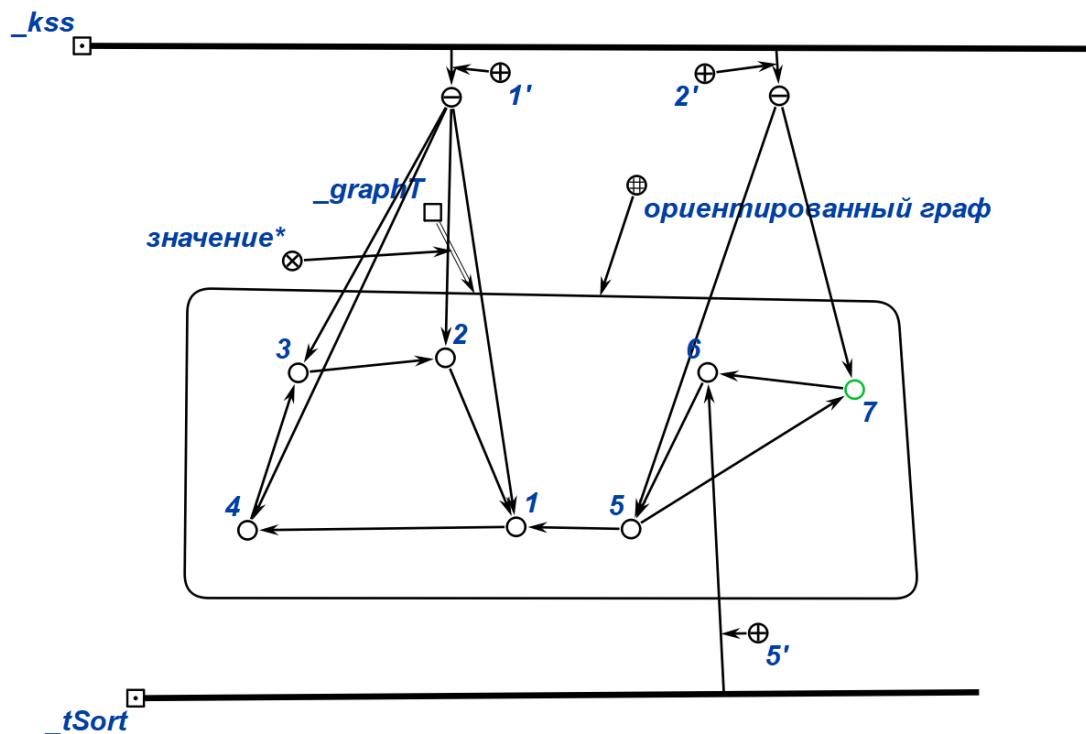


g. Т.к. от вершины 4 нет дуг к вершинам, присутствующим в списке топологической сортировки, возвращаемся к вершине 1.

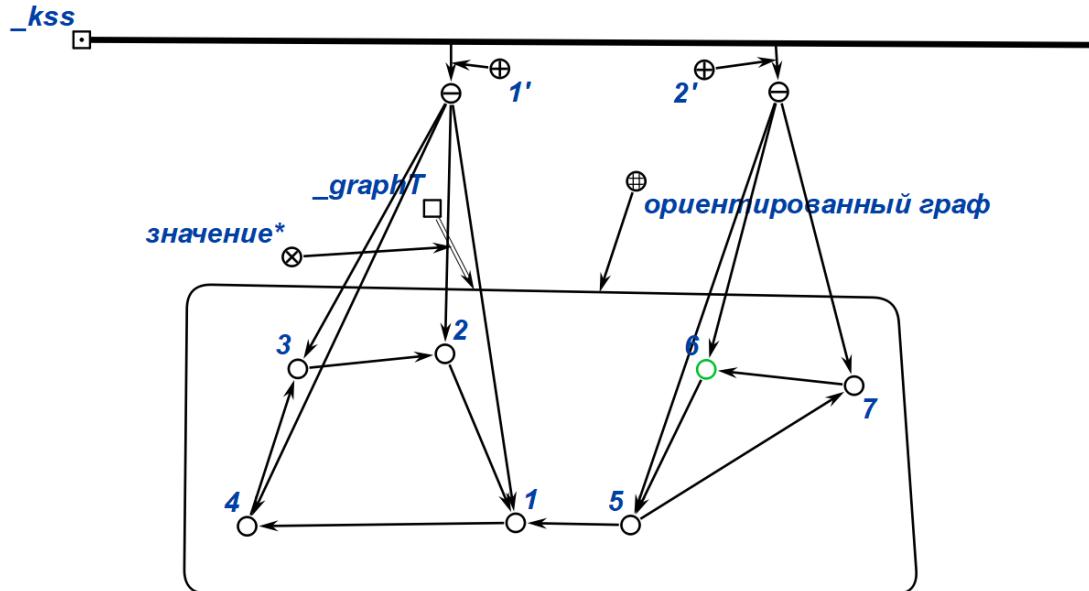


h. Т.к. от вершины 1 нет дуг к вершинам, присутствующим в списке топологической сортировки, первый обход транспони-

рованного графа закончен. Начинаем второй обход с вершины, на данный момент имеющей максимальный номер в списке топологической сортировки, - 5. Создаём пустое множество, обозначающее вершины графов  $_graph$  и  $_graphT$ , относящиеся ко второй компоненте сильной связности. Добавляем вершину 5 в множество вершин второй компоненты сильной связности в  $_kss$  и удаляем её же из  $_tSort$ .

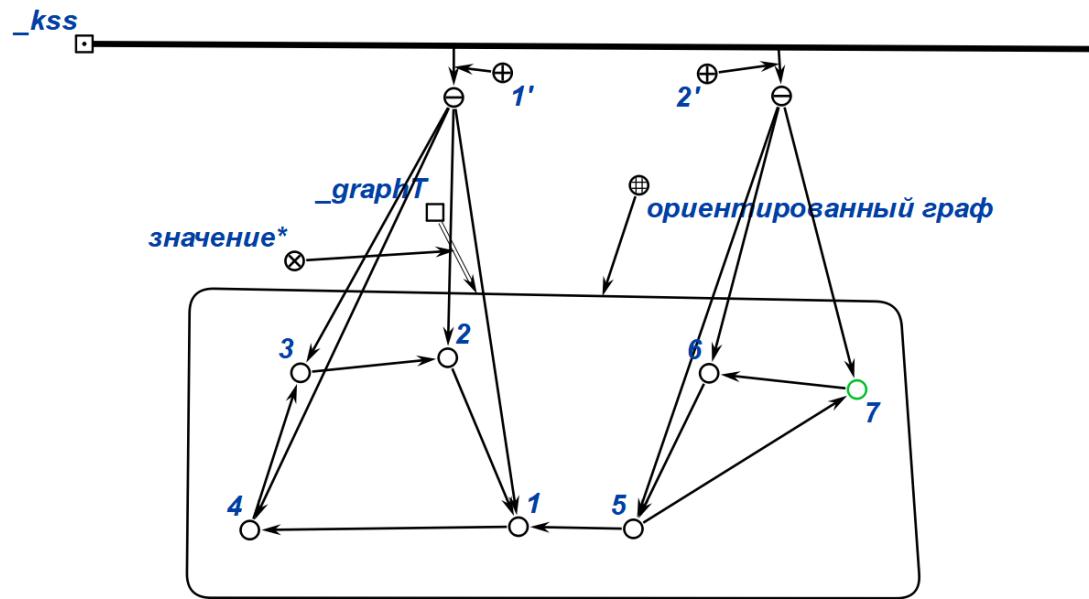


- Переходим к вершине с максимальным номером в  $_tSort$ , к которой от вершины 5 есть дуга - 7. Добавляем вершину 7 в множество вершин второй компоненты сильной связности в  $_kss$  и удаляем её же из  $_tSort$ .



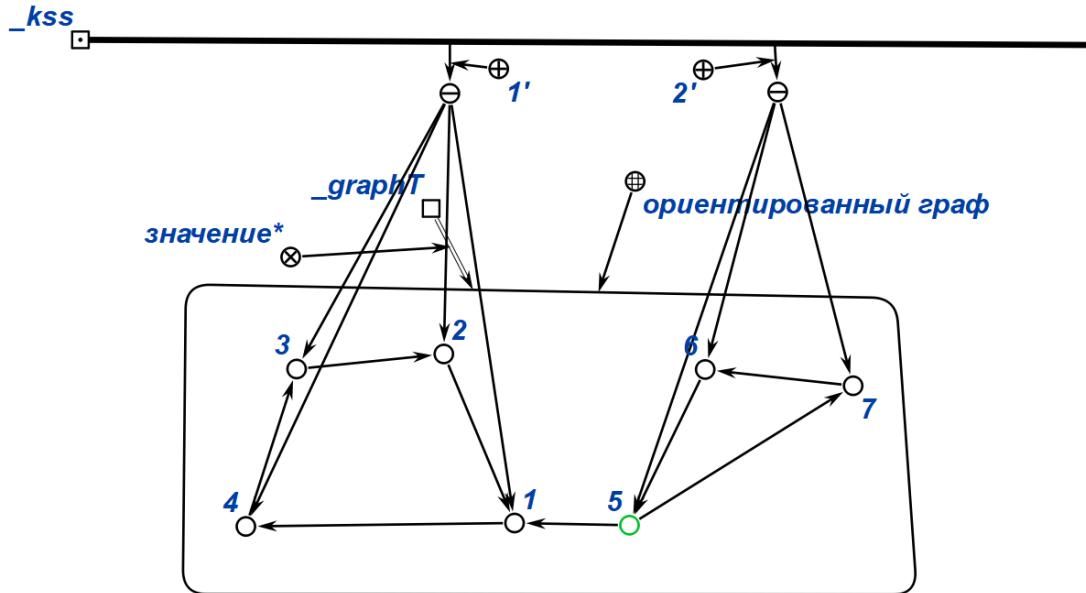
`_tSort`

- j. Переходим к вершине с максимальным номером в `_tSort`, к которой от вершины  $7$  есть дуга -  $6$ . Добавляем вершину  $6$  в множество вершин второй компоненты сильной связности в `_kss` и удаляем её же из `_tSort`.



`_tSort`

- k. Т.к. от вершины  $6$  нет дуг к вершинам, присутствующим в списке топологической сортировки, возвращаемся к вершине  $7$ .



1. Т.к. от вершины 7 нет дуг к вершинам, присутствующим в списке топологической сортировки, возвращаемся к вершине 5. Т.к. от вершины 5 нет дуг к вершинам, присутствующим в списке топологической сортировки, второй обход транспонированного графа закончен. Список топологической сортировки пуст, так что построение списка компонент сильной связности завершено.

## 6. Построение графа конденсации.

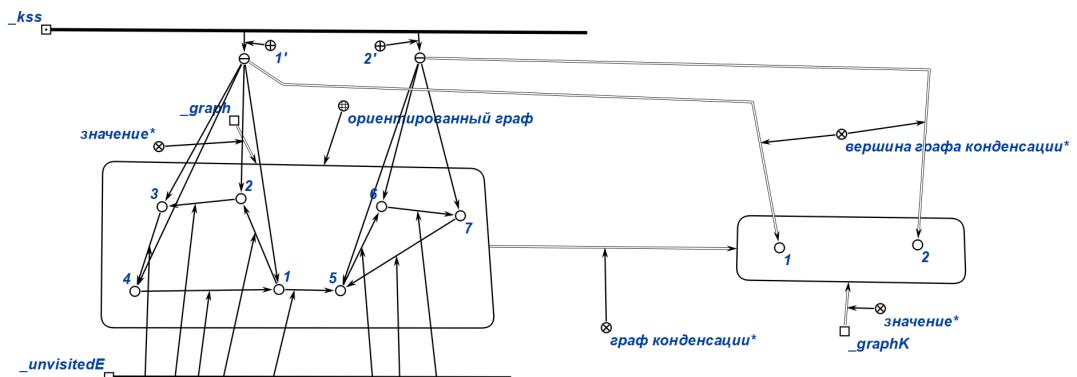
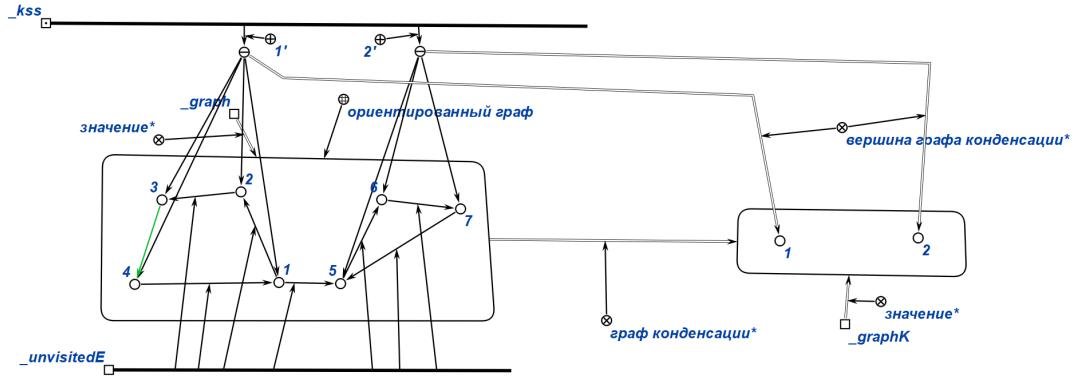
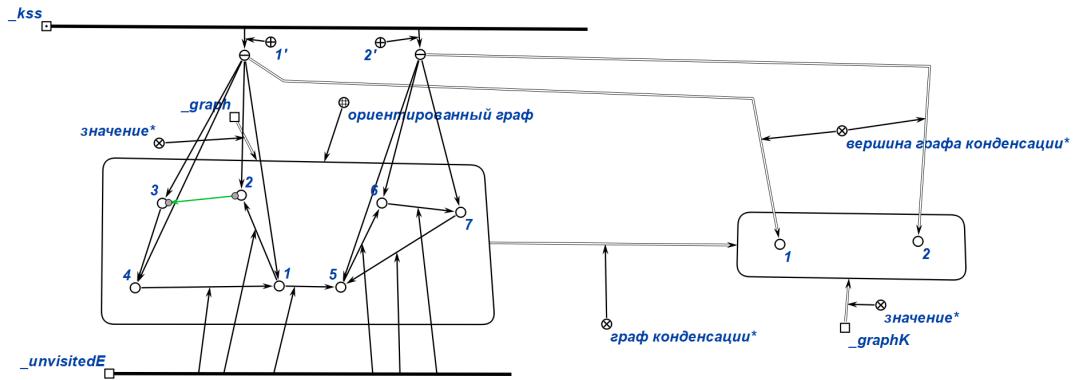


Рисунок 4.6 – Шаг 6

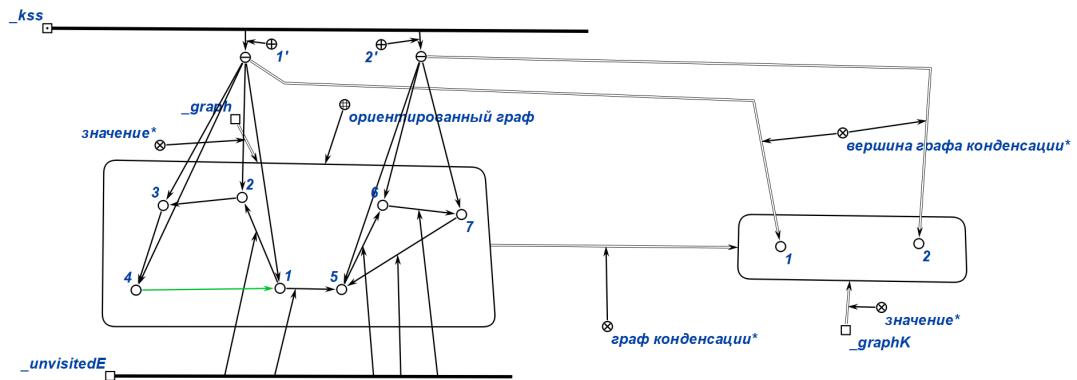
Создаётся пустой граф  $_graphK$ . Т.к. в списке компонент сильной связности два множества, добавляем вершины 1 и 2 в  $_graphK$ . Создаём множество  $_unvisitedE$ , в котором содержатся все дуги  $_graph$ .



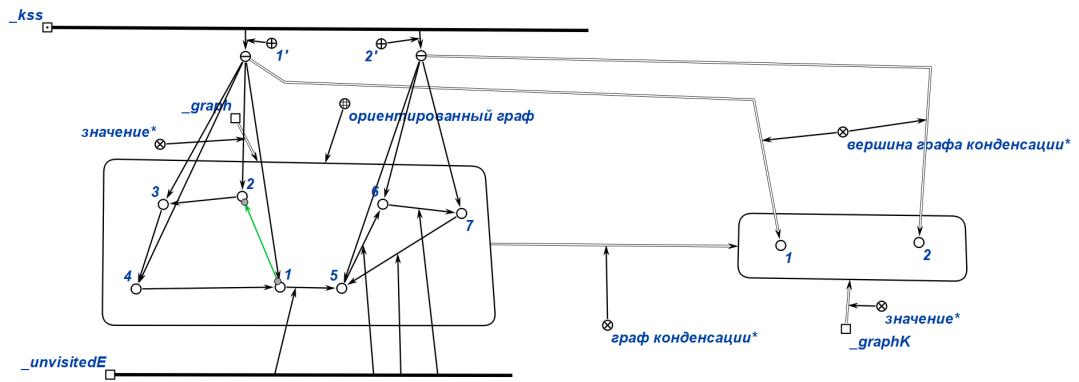
a. Дуга  $3 \rightarrow 4$  связывает вершины из первой компоненты сильной связности. Удаляем её из  $_unvisitedE$ .



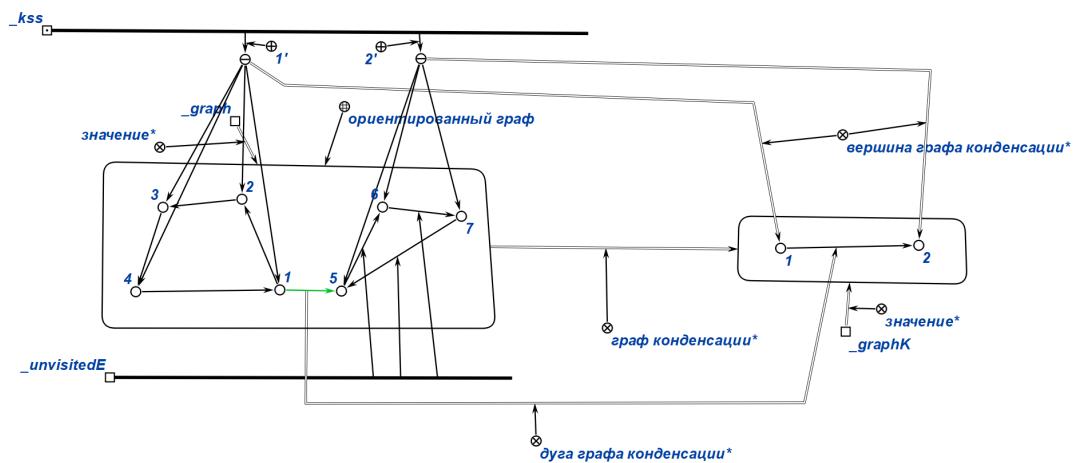
b. Дуга  $2 \rightarrow 3$  связывает вершины из первой компоненты сильной связности. Удаляем её из  $_unvisitedE$ .



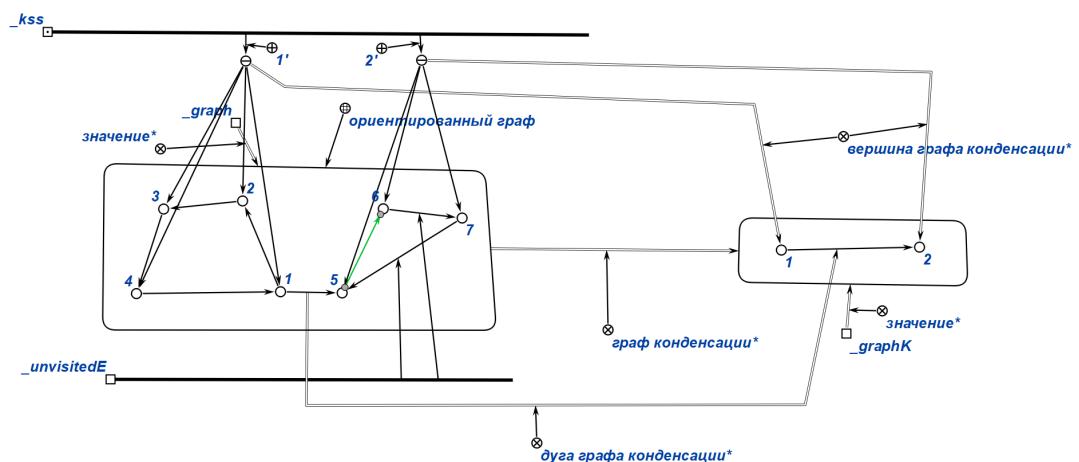
c. Дуга  $4 \rightarrow 1$  связывает вершины из первой компоненты сильной связности. Удаляем её из  $_unvisitedE$ .



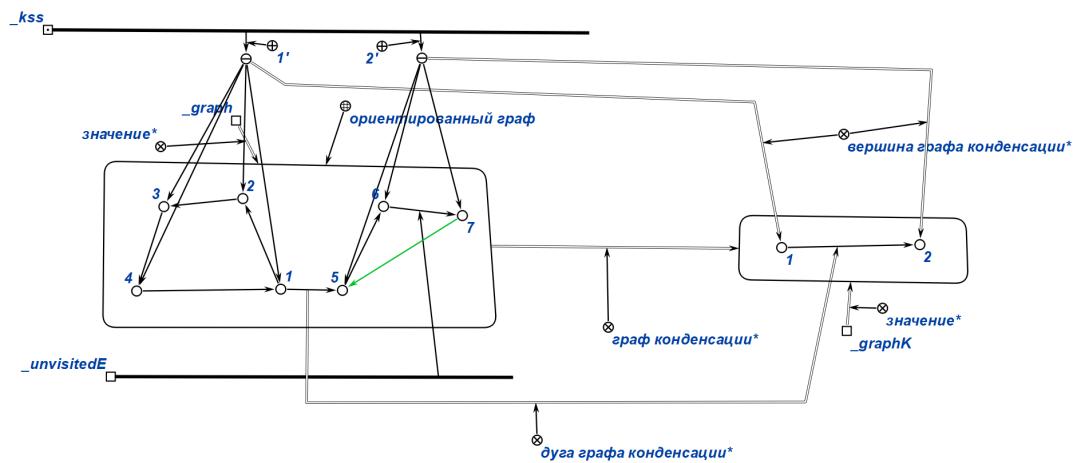
d. Дуга  $1 \rightarrow 2$  связывает вершины из первой компоненты сильной связности. Удаляем её из  $_unvisitedE$ .



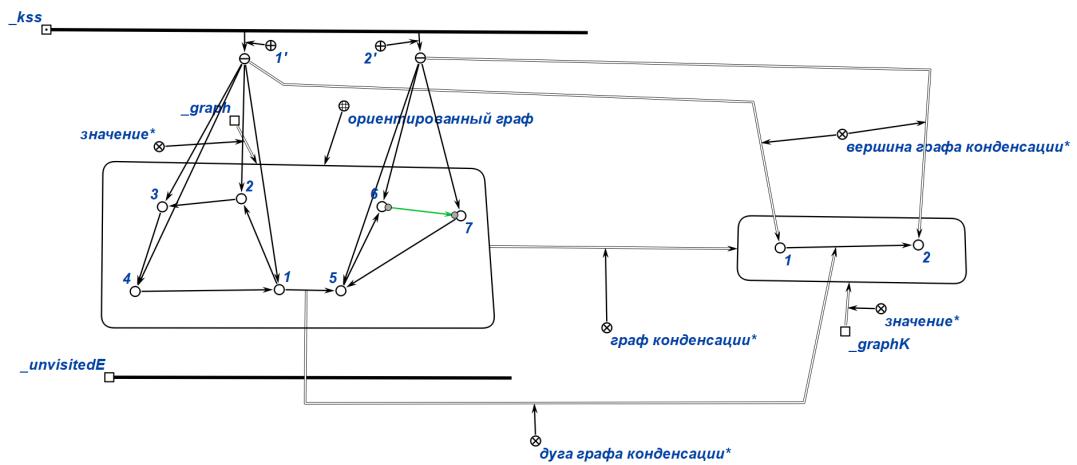
e. Дуга  $1 \rightarrow 5$  связывает вершины из первой и второй компонент сильной связности соответственно, поэтому записываем дугу  $1 \rightarrow 2$  в  $_graphK$ . Удаляем дугу  $1 \rightarrow 5$  из  $_unvisitedE$ .



f. Дуга  $5 \rightarrow 6$  связывает вершины из второй компоненты сильной связности. Удаляем её из  $_unvisitedE$ .



g. Дуга  $\gamma \rightarrow 5$  связывает вершины из второй компоненты сильной связности. Удаляем её из  $_unvisitedE$ .



h. Дуга  $6 \rightarrow 7$  связывает вершины из второй компоненты сильной связности. Удаляем её из  $_unvisitedE$ . Т.к. непосещённых дуг в графе не осталось, построение графа конденсации завершено.

7. Вывод ответа: Выводим граф  $_graphK$ , который является графом конденсации для заданного графа  $_graph$ .

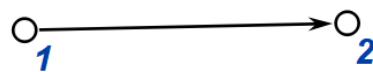


Рисунок 4.7 – Вывод

## **5 ЗАКЛЮЧЕНИЕ**

Был разработан и формализован при помощи SCg-кода алгоритм нахождения компонент сильной связности и построения на их основе графа конденсации для заданного ориентированного графа.

Кроме того, были получены общие навыки работы в графовом редакторе KBE и перевода текстовой информации в графовую структуру, состоящую из вершин, ориентированных и неориентированных множеств, связок, ориентированных рёбер (дуг) и контуров.