pgRouting Contraction

Rohith Reddy

GSoC 2016

Contraction

- Graph Contraction, when working on big graphs:
  - road graphs,
  - electric networks
- Speeds up some graph algorithms.
- The current implementation:
  - Flexible Framework
    - “Easy” to add a new operation.
  - Dead end contraction
  - Linear contraction

The contraction skeleton

- In general we have an initial set up that may involve analyzing the graph given as input and setting the non contractible nodes or edges.
- We have a cycle that will go and perform a contraction operation until while possible, and then move to the next contraction operation.
- Adding a new operation then becomes an “easy” task but more things might be involved, because the characteristics of the graph change each time its contracted, so some interaction between contractions has to be implemented also.
- Currently, there are two implemented operation for contracting a graph
    - Dead End contraction
    - Linear contraction

Dead End Contraction

Dead End Node

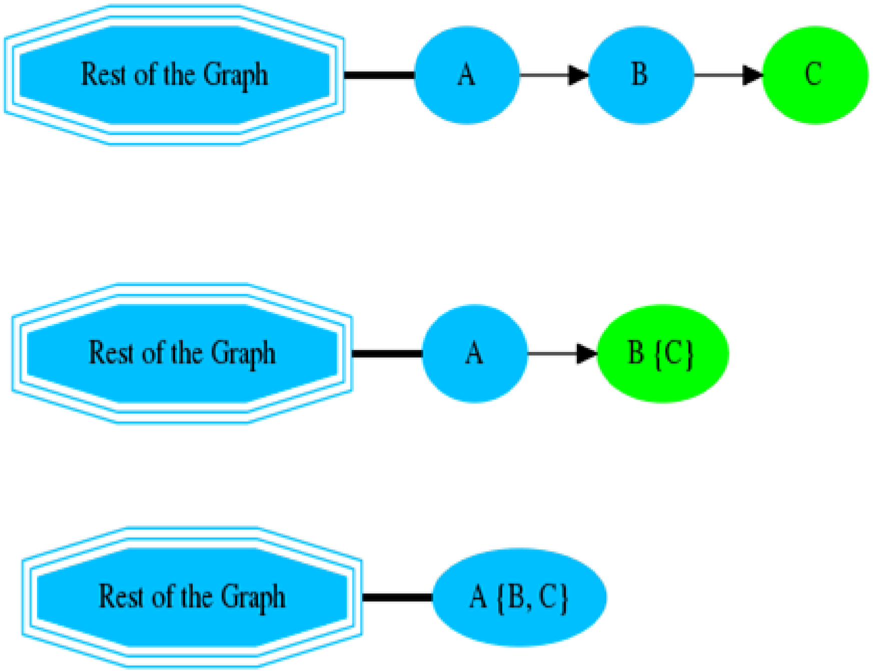
Undirected Graph

- The number of adjacent vertices is one.

Directed Graph

- Case 1
  - No outgoing edges
  - At least one incoming edge.
- Case 2
  - One incoming edge
  - One outgoing edge
  - Same identifier on the edges.

Dead End Contraction

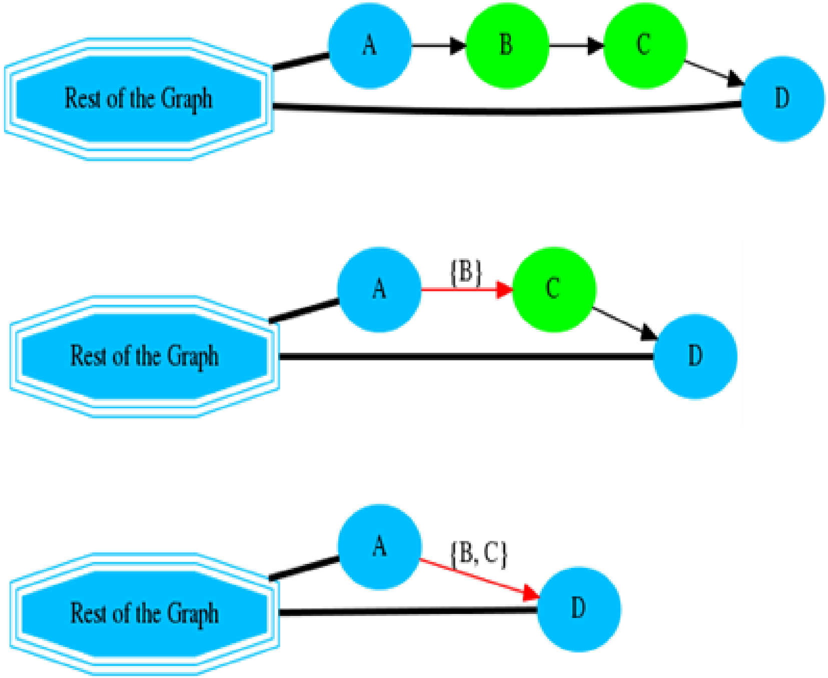


Linear Contraction

Linear Node

- Two adjacent vertices.
- At least one incoming edge and one outgoing edge.

Linear Contraction

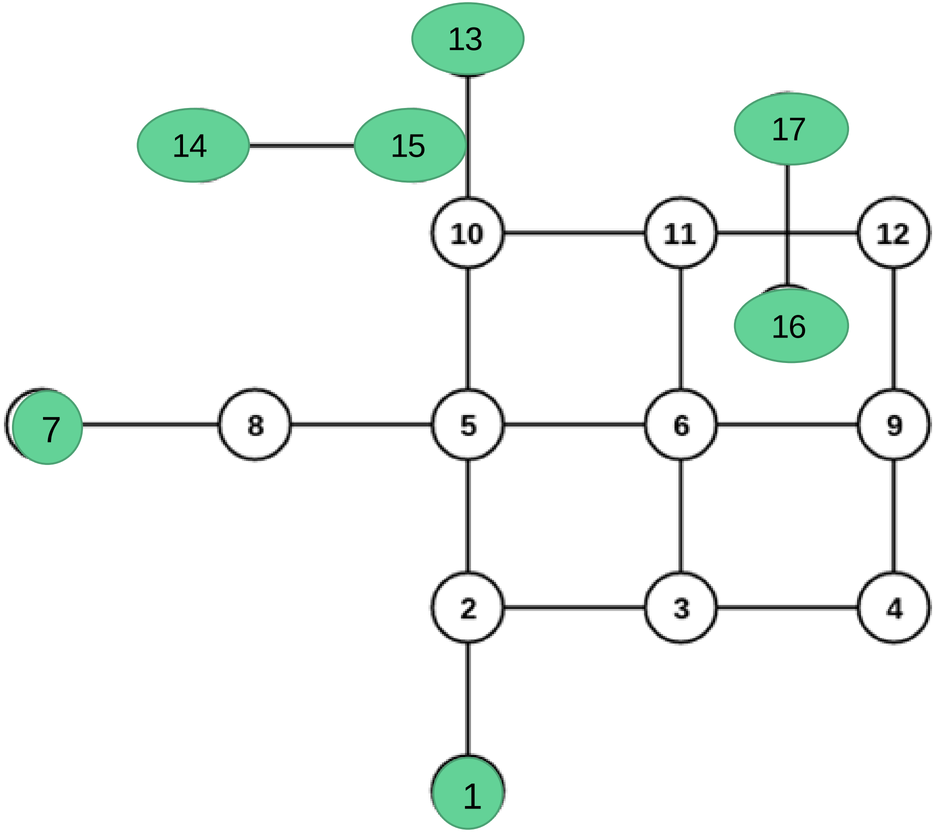


Example

Using the Sample Data of the:

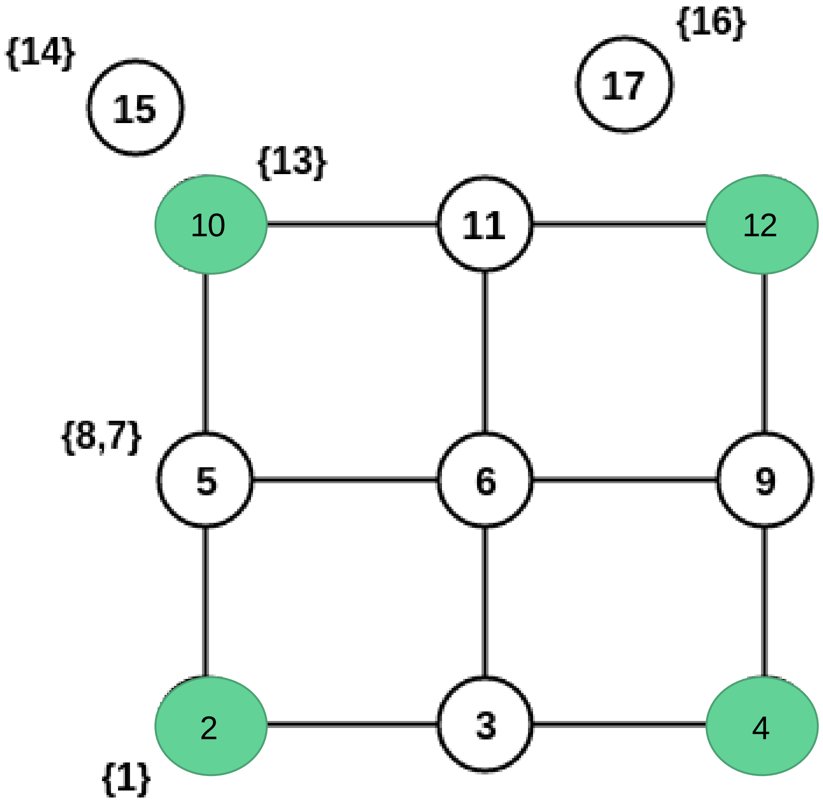
pgRouting documentation

Original Graph



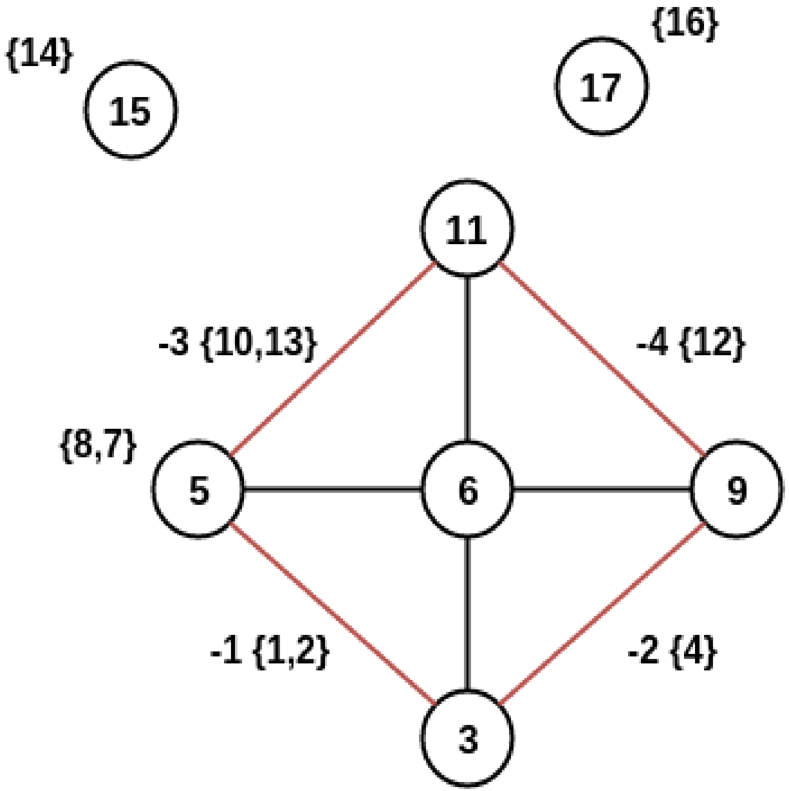
Dead end nodes

After Dead End Contraction



Linear nodes

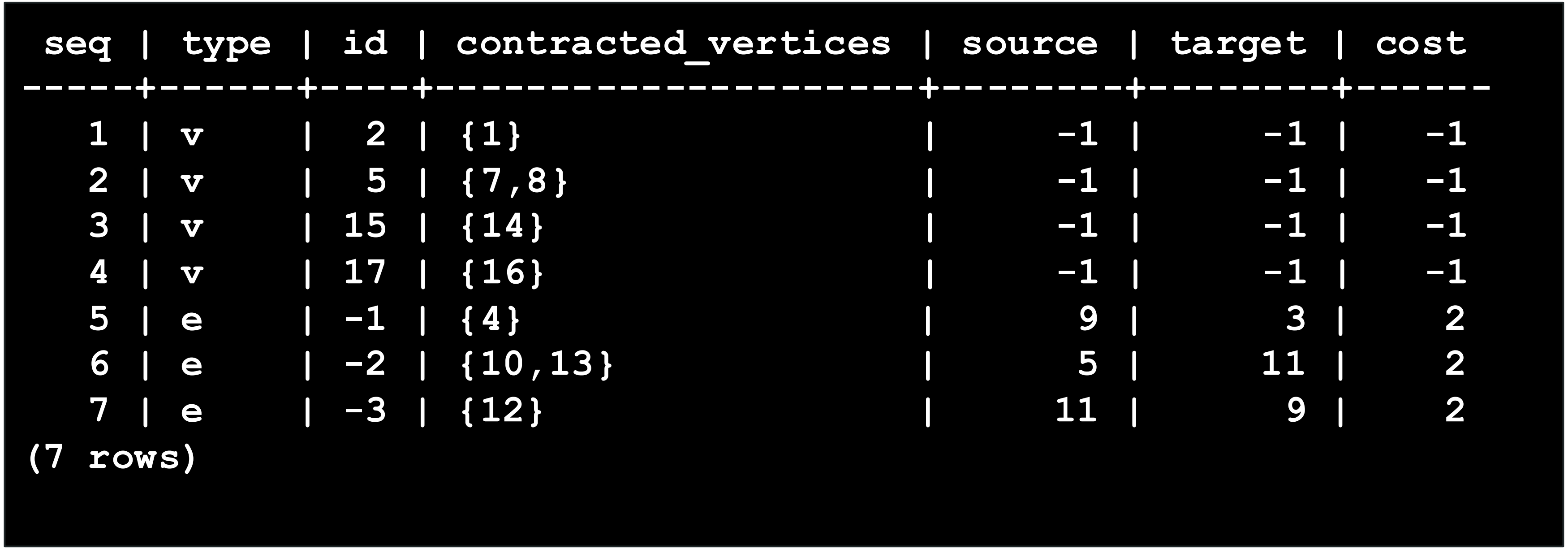
After Linear Contraction



The Query

```
SELECT *
FROM pgr_contractGraph(
    'SELECT id,
        source, target,
        cost, reverse_cost
     FROM edge_table',
     ARRAY[1, 2])
```

Query Results



```
 seq | type | id | contracted_vertices | source | target | cost
-----+------+----+---------------------+--------+--------+------
   1 | v    |  2 | {1}                 |     -1 |     -1 |   -1
   2 | v    |  5 | {7,8}               |     -1 |     -1 |   -1
   3 | v    | 15 | {14}                |     -1 |     -1 |   -1
   4 | v    | 17 | {16}                |     -1 |     -1 |   -1
   5 | e    | -1 | {4}                 |      9 |      3 |    2
   6 | e    | -2 | {10,13}             |      5 |     11 |    2
   7 | e    | -3 | {12}                |     11 |      9 |    2
(7 rows)
```