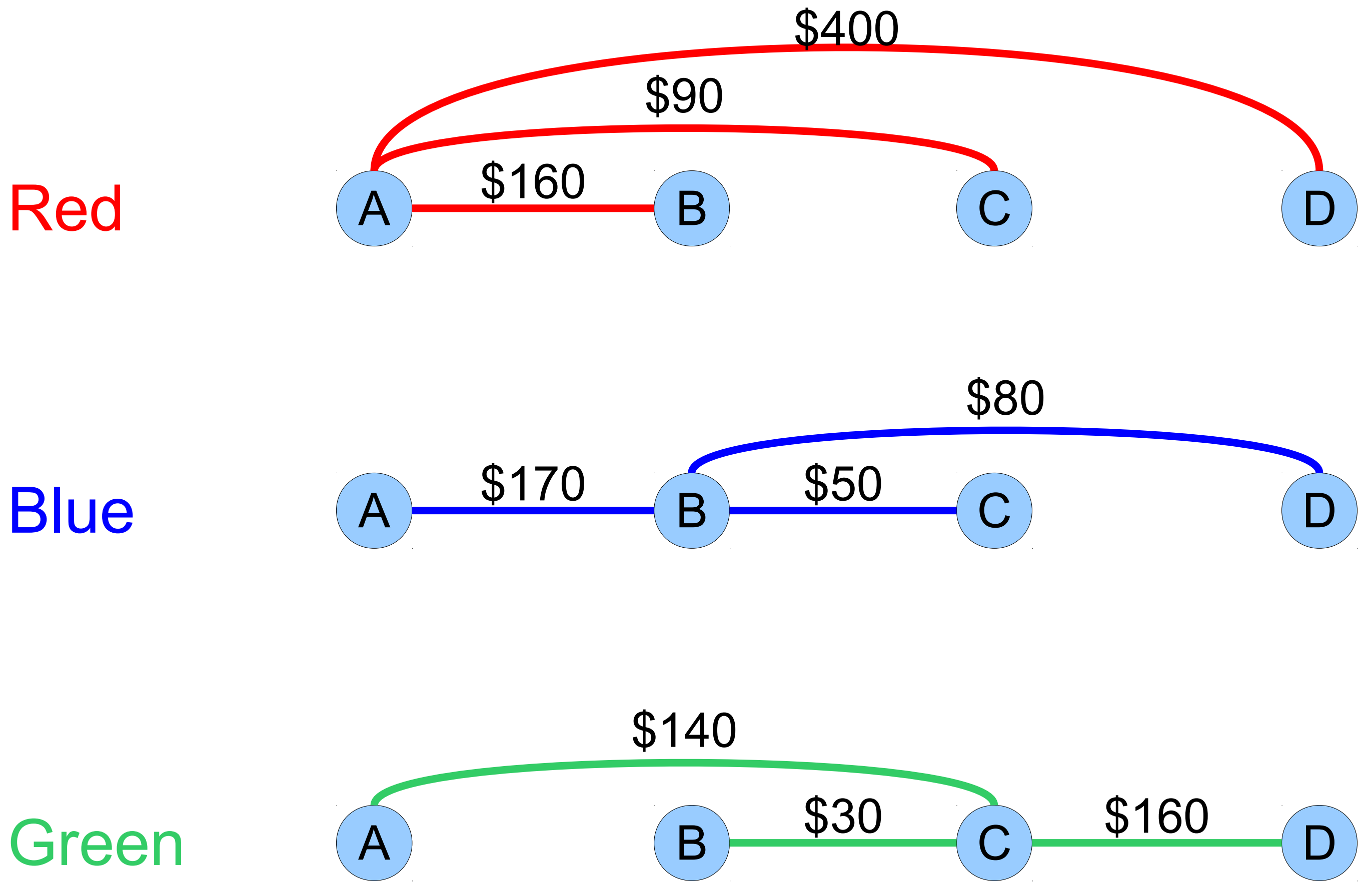
Examples of Shortest Path Problems

Dijkstra's algorithm
Bellman-Ford algorithm
Modeling

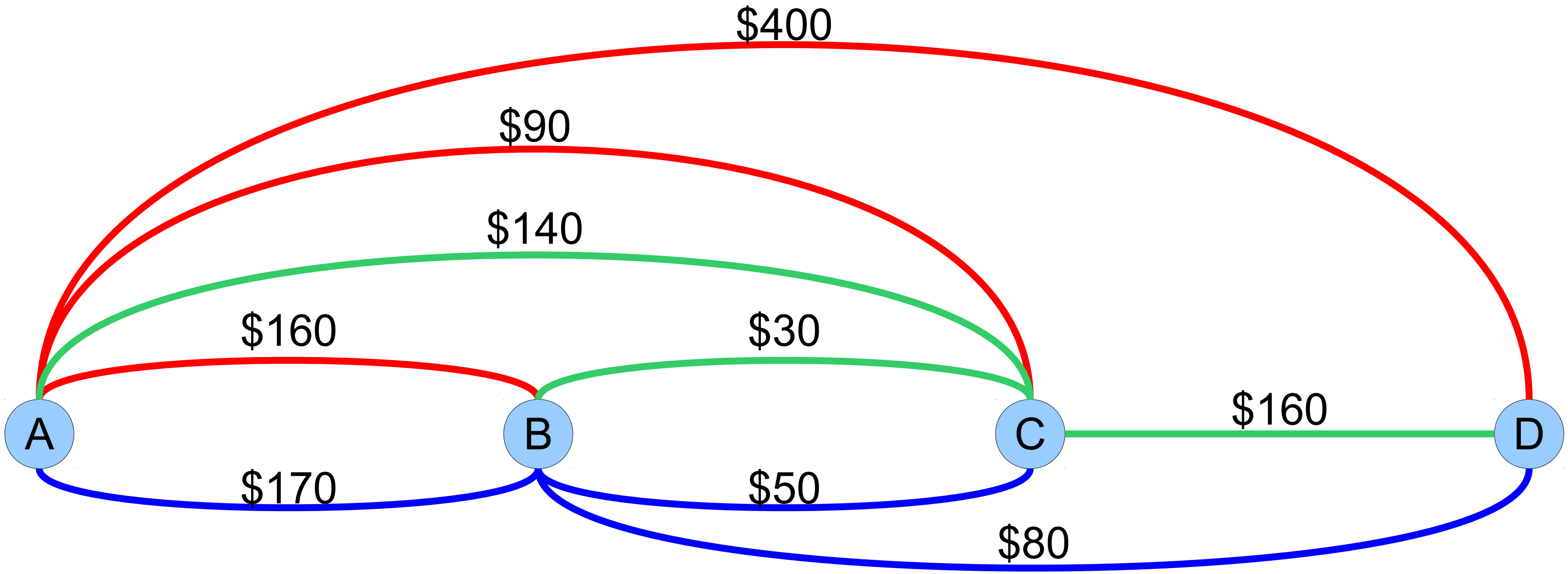
Trip Planning

4 cities: A, B, C, D;   3 airliners: Red, Blue, Green;
Cheapest way of going from A to D ?



First Try

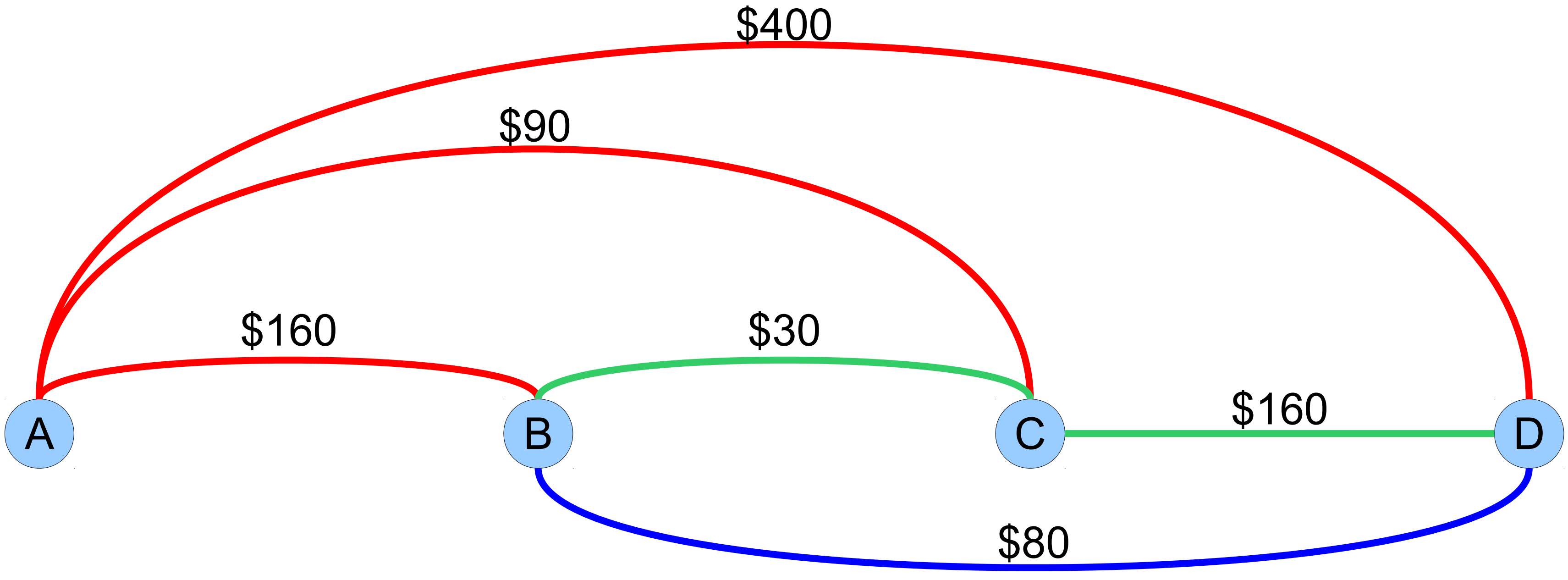
Vertices: 4 cities { A, B, C, D }
Edges: flights, undirected, weights – prices
Objective: shortest path from A to D (min total weight/cost)



Multiple edges between some vertices ! (Multi-graph)

Working on Simple Graph

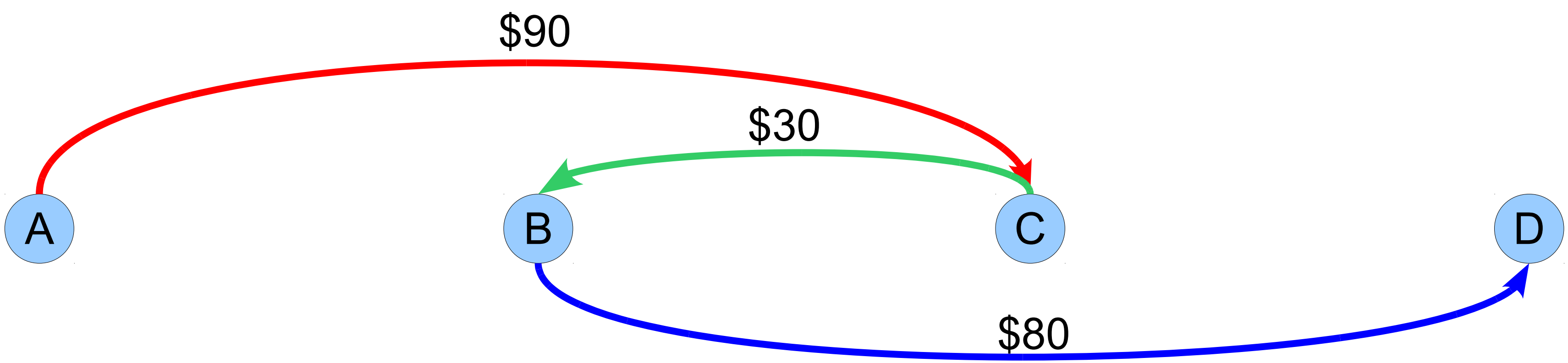
Vertices: 4 cities { A, B, C, D }
Edges: flights, undirected, weights – prices
Objective: shortest path from A to D (min total weight/cost)



Only keep the cheaper flights
Dijkstra? Bellman-Ford?

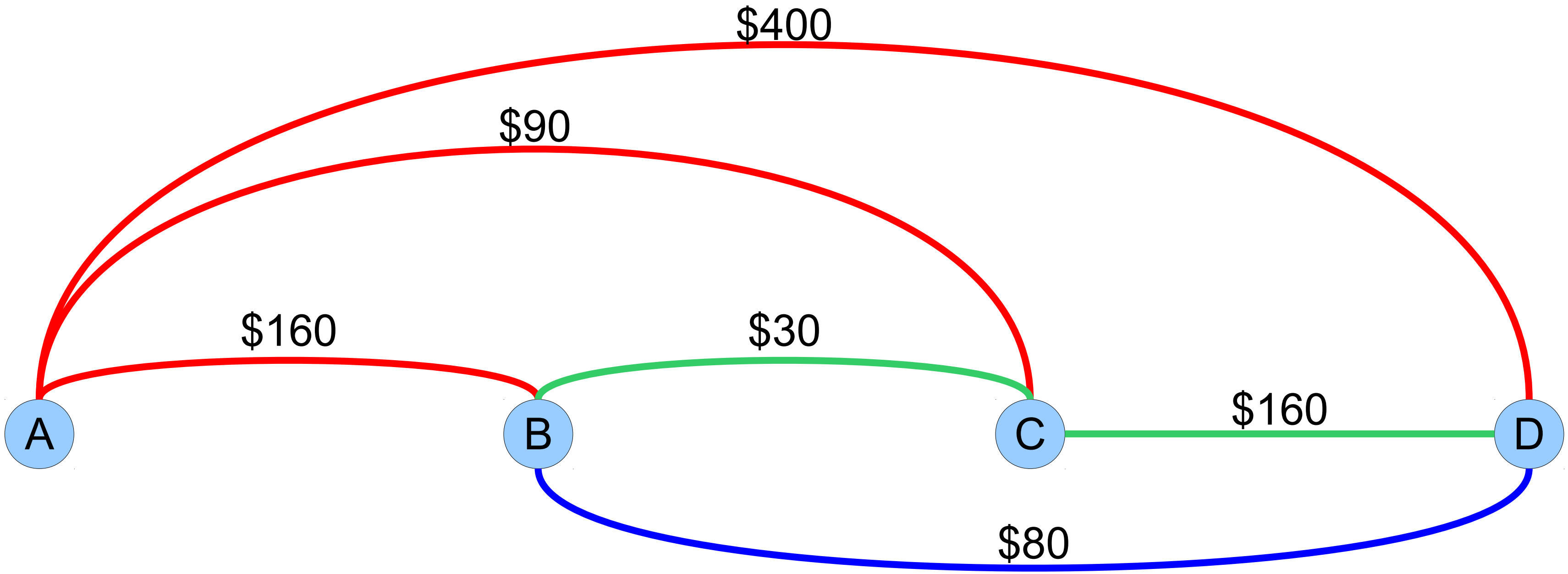
Fewer Transfers

It is cheap ($200), but have to transfer twice ...
You can tolerate at most 1 transfer, what is the cheapest way?



Fewer Transfers

It is cheap ($200), but have to transfer twice ...
You can tolerate at most 1 transfer, what is the cheapest way?



The k'th iteration of Bellman-Ford gives the shortest paths with length at least k

Modified Bellman-Ford

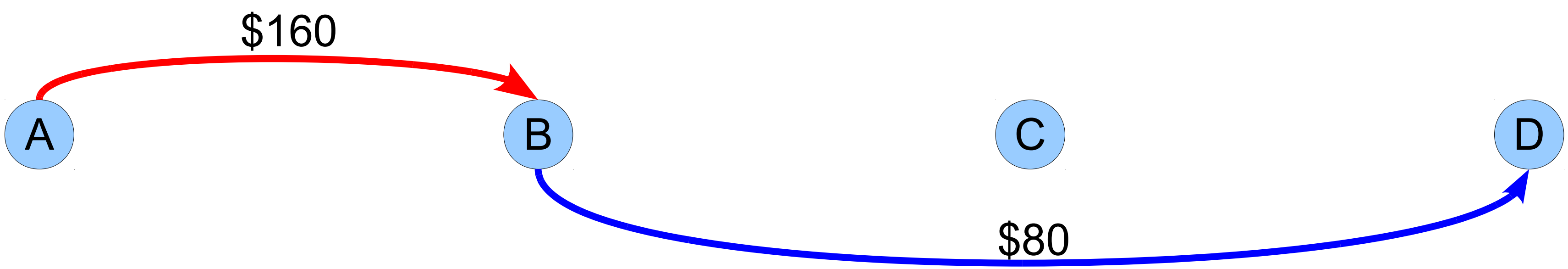
An algorithm to find the shortest path with at most k edges:
G - directed graph;   s - start vertex;   t - end vertex

```
SHORTEST-PATH-WITH-AT-MOST-K-EDGES (G, s, t)
    d[0][s] = 0
    d[0][v] = ∞ for v ∈ V and v = s

    for i = 1..k do
       d[k][v] = d[k − 1][v]
       for any (u, v) ∈ E do
          d[k][v] = min(d[k][v], d[k − 1][u] + w[u][v])

    return d[k][t]
```

Fewer Transfers

The cheapest way of going from A to B with at most 2 transfers: $240

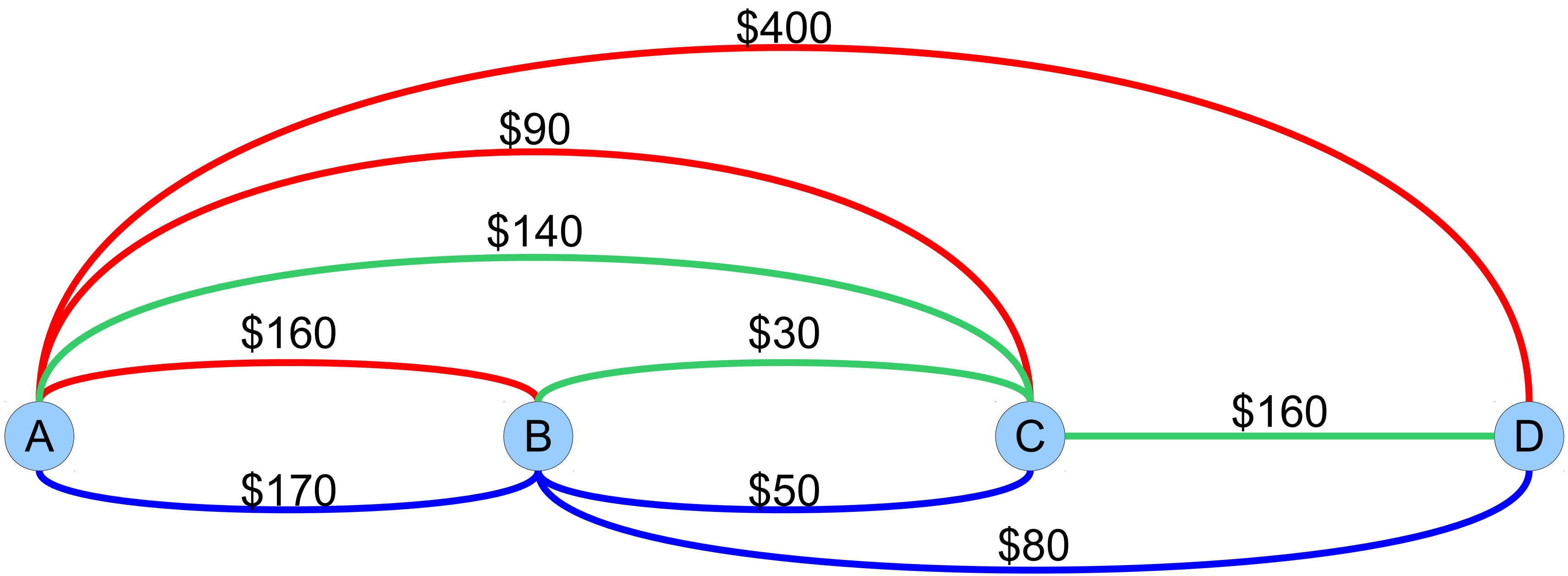
Only 2 iterations in Bellman-Ford algorithm are needed



Wait! I Have a Coupon!

Have a $100 coupon :

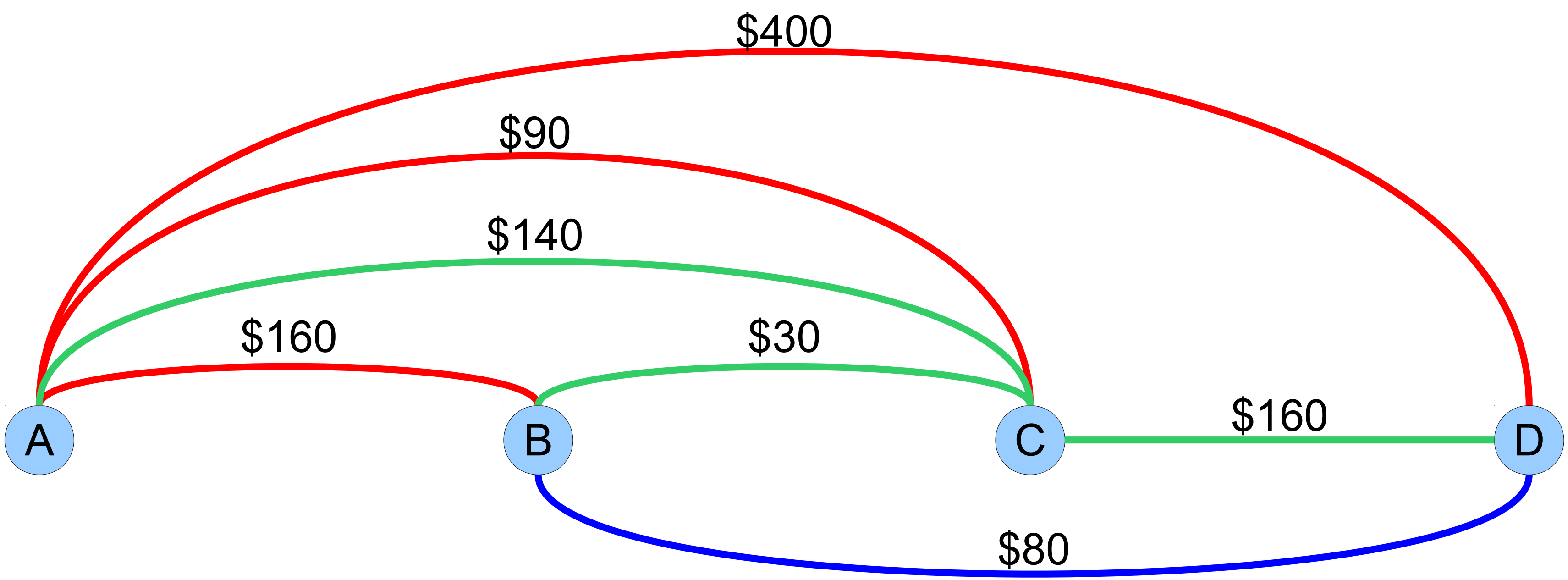
- Can be used with any airliner;
- Can be used only once.
- Can NOT be used on a flight whose price is smaller than the coupon value



Wait! I Have a Coupon!

Have a $100 coupon :

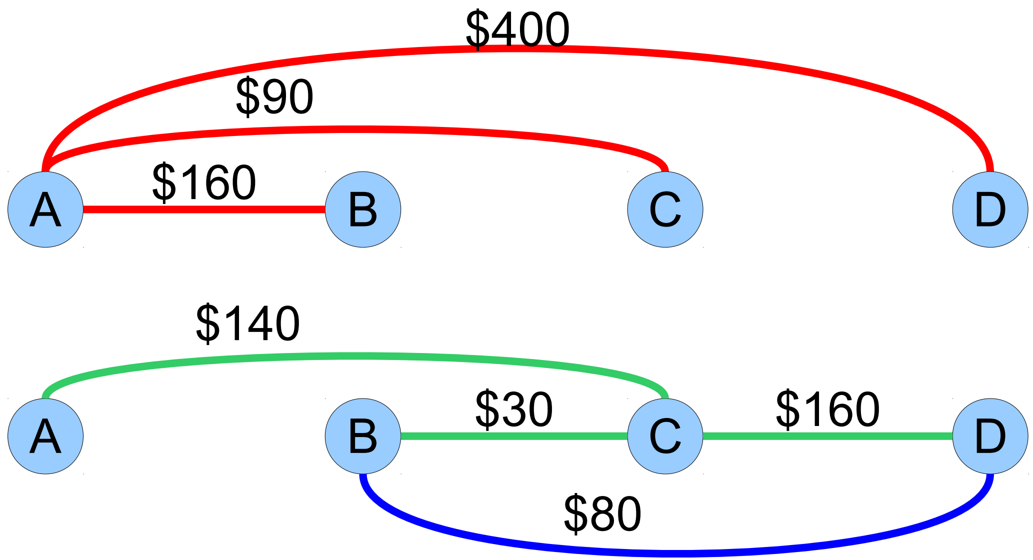
- Can be used with any airliner;
- Can be used only once.
- Can NOT be used on a flight whose price is smaller than the coupon value



Mutiple flights between A and C !

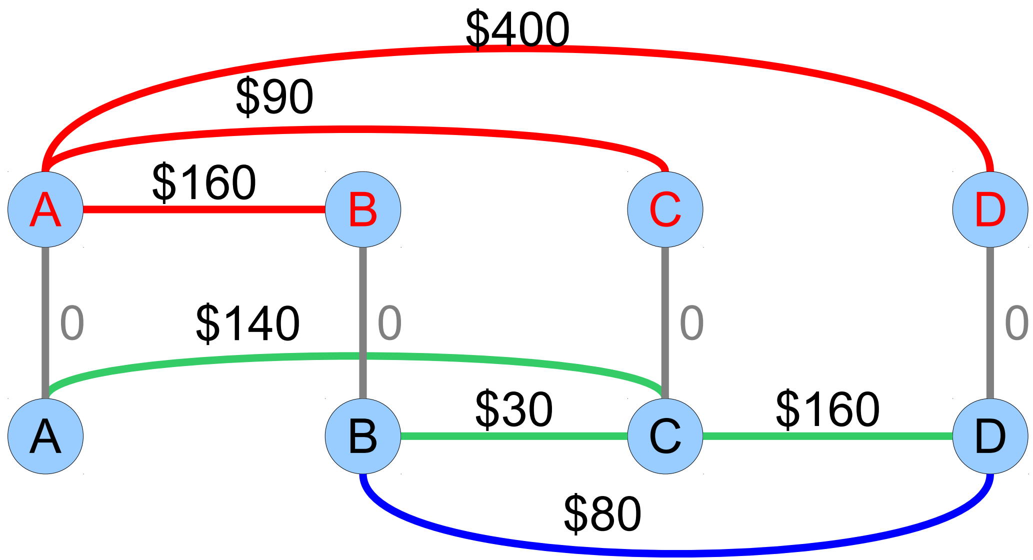
Modeling

- Separate Red flights from the others



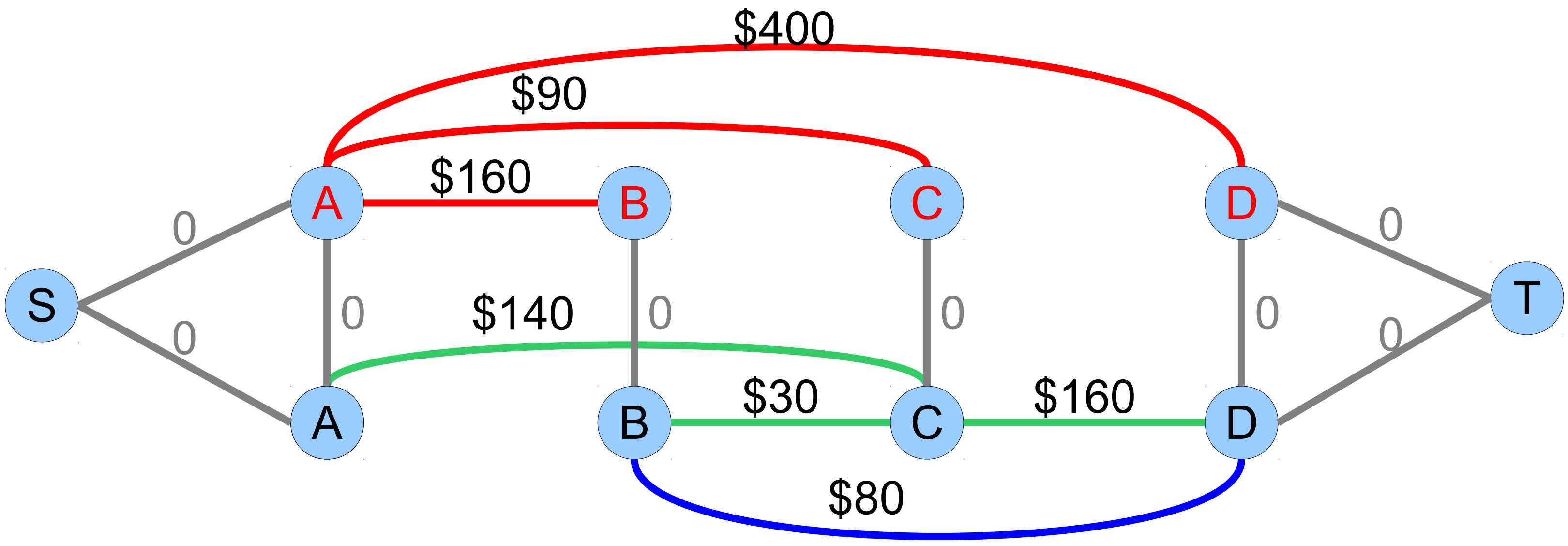
Modeling

- Separate Red flights from the others
- Transfer cost is 0



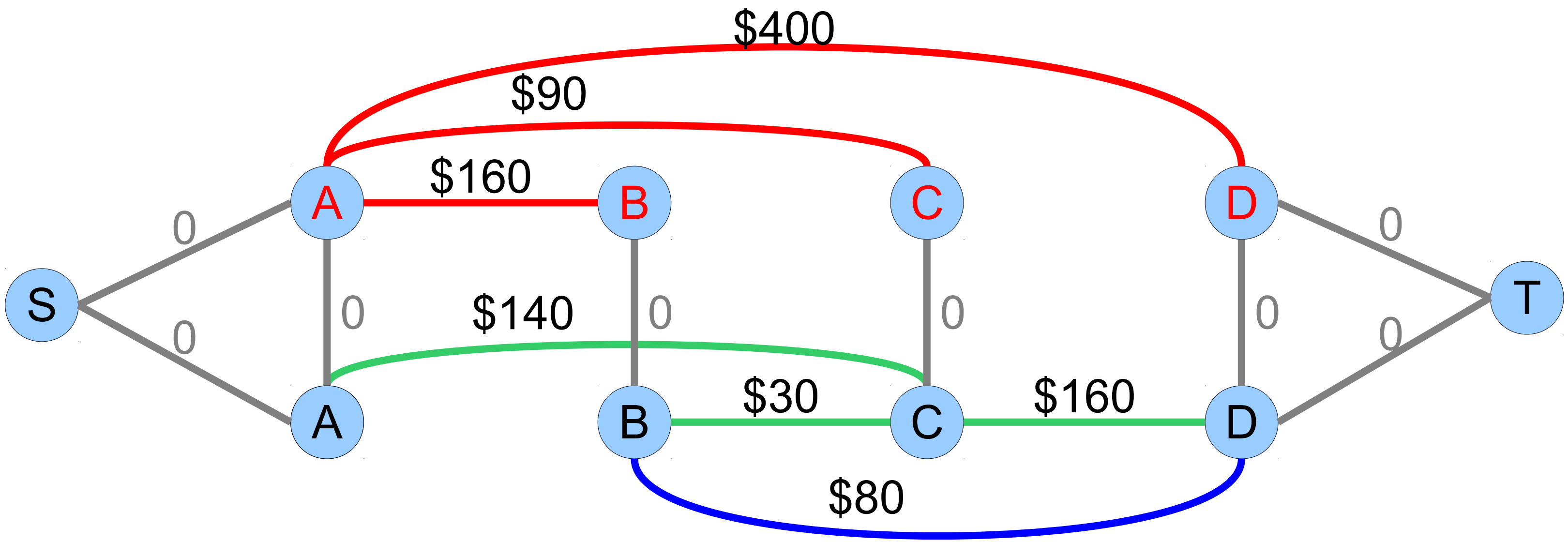
Modeling

- Separate Red flights from the others
- Transfer cost is 0
- Add artificial source and destination (with 0-weight edges)

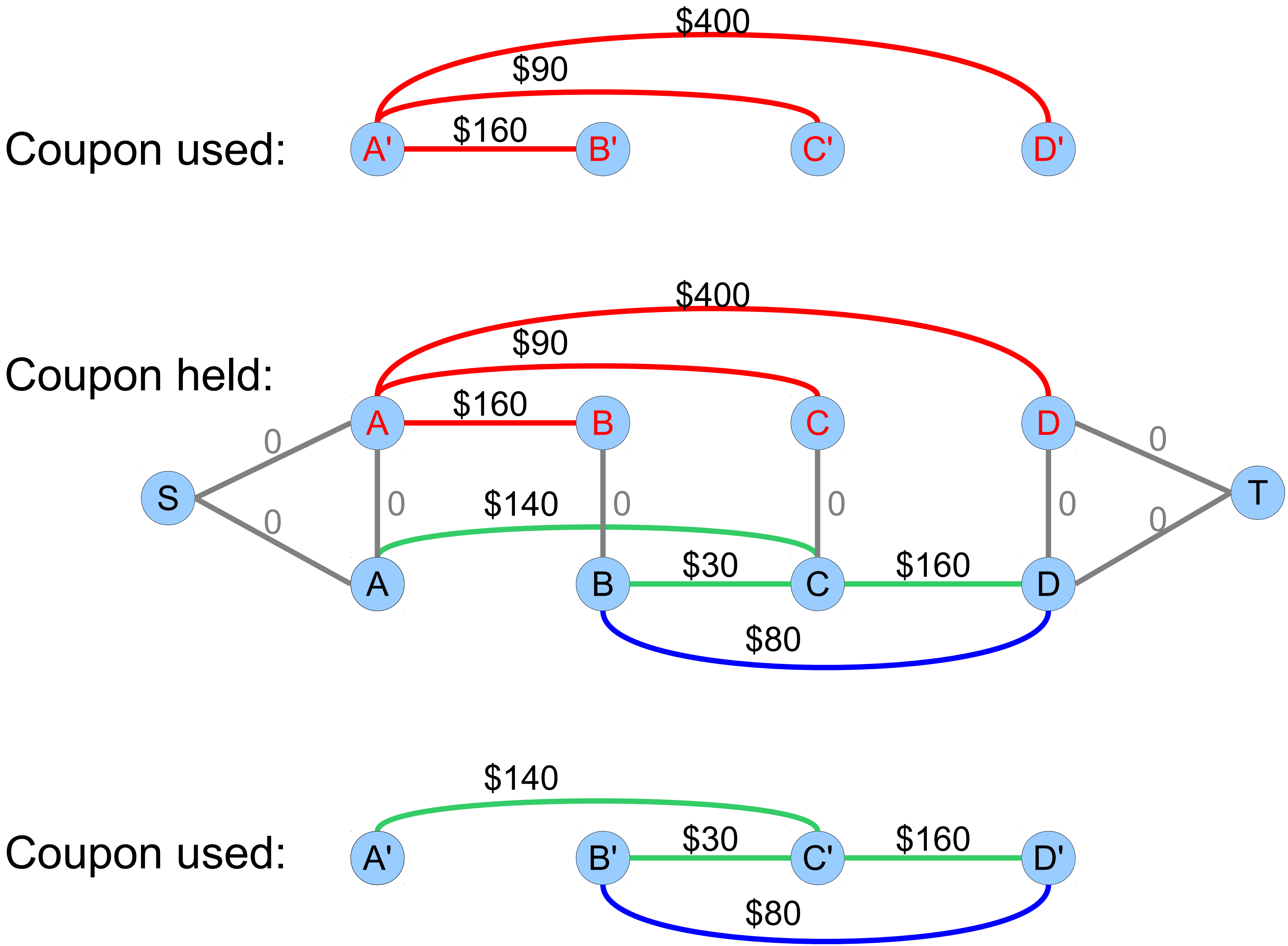


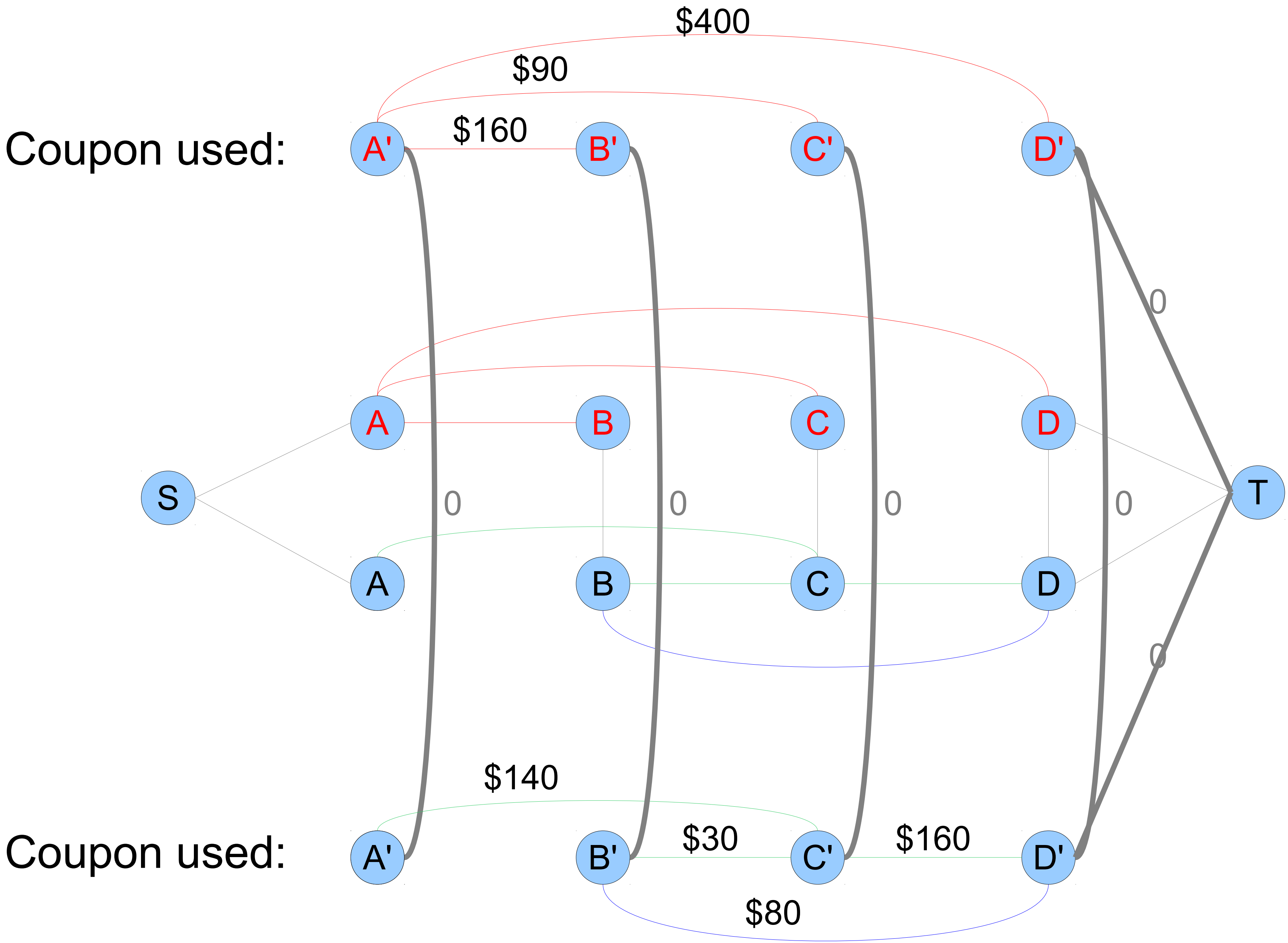
Modeling

- Separate Red flights from the others
- Transfer cost is 0
- Add artificial source and destination (with 0-weight edges)

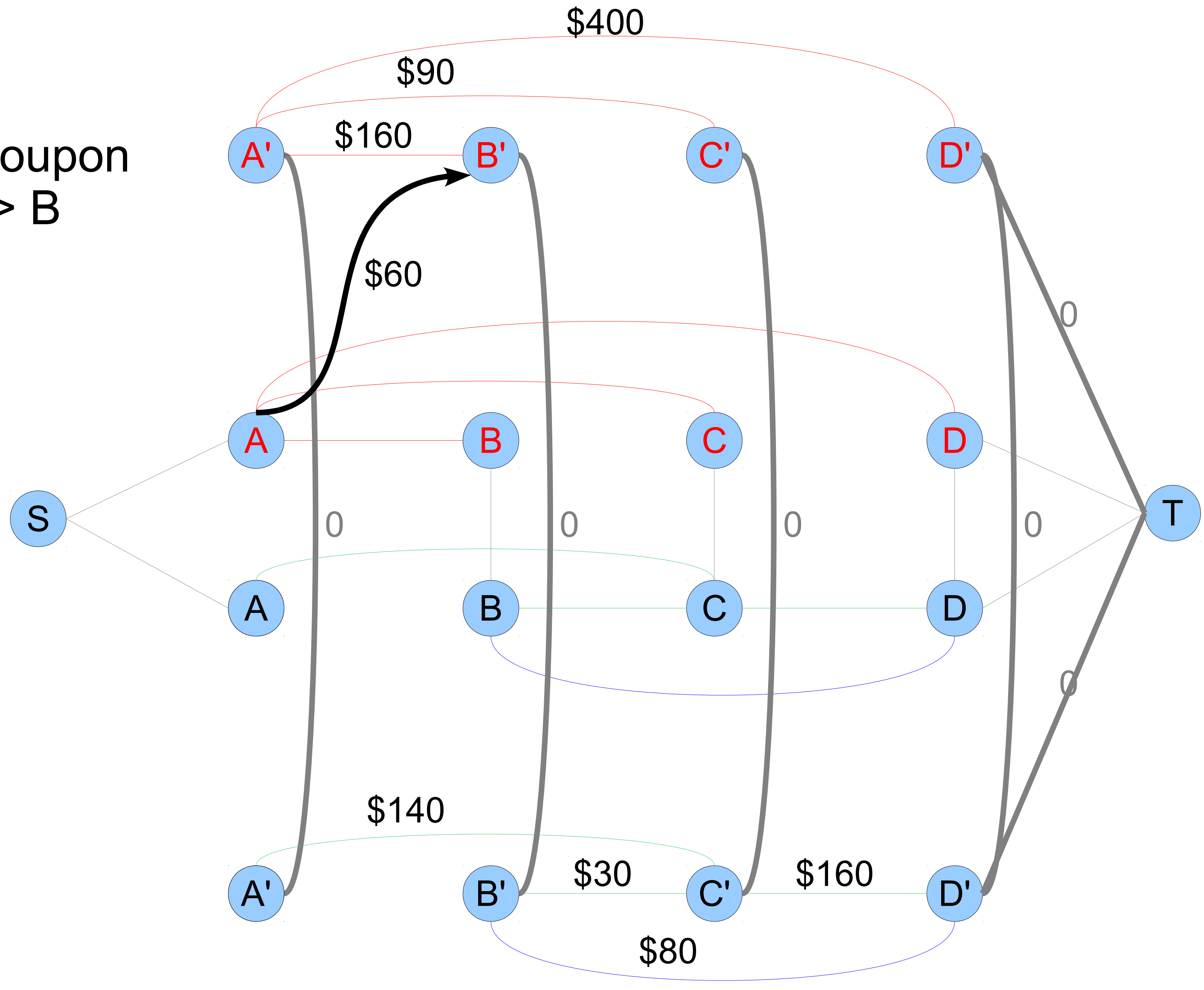


How about the coupon?

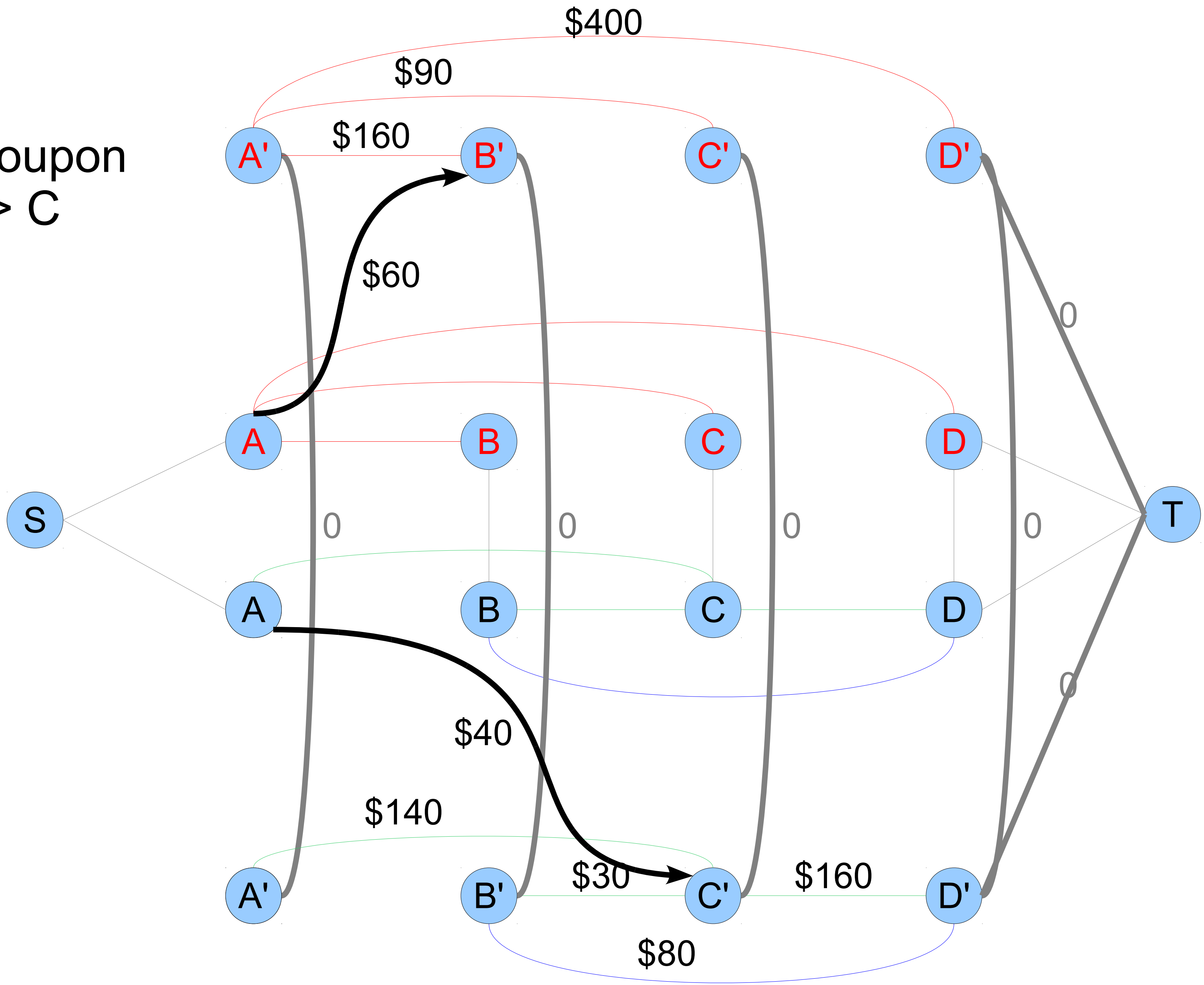

Coupon used:
$400
$90
$160
A'
B'
C'
D'
Coupon held:
$400
$90
$160
A
B
C
D
S
T
0
0
0
0
0
0
0
0
$140
$30
$160
$80
A
B
C
D
Coupon used:
$140
$30
$160
$80
A'
B'
C'
D'


$400
$90
$160
Coupon used:
A'
B'
C'
D'
0
A
B
C
D
S
T
0
0
0
0
A
B
C
D
0
$140
Coupon used:
A'
$30
$160
B'
C'
D'
$80

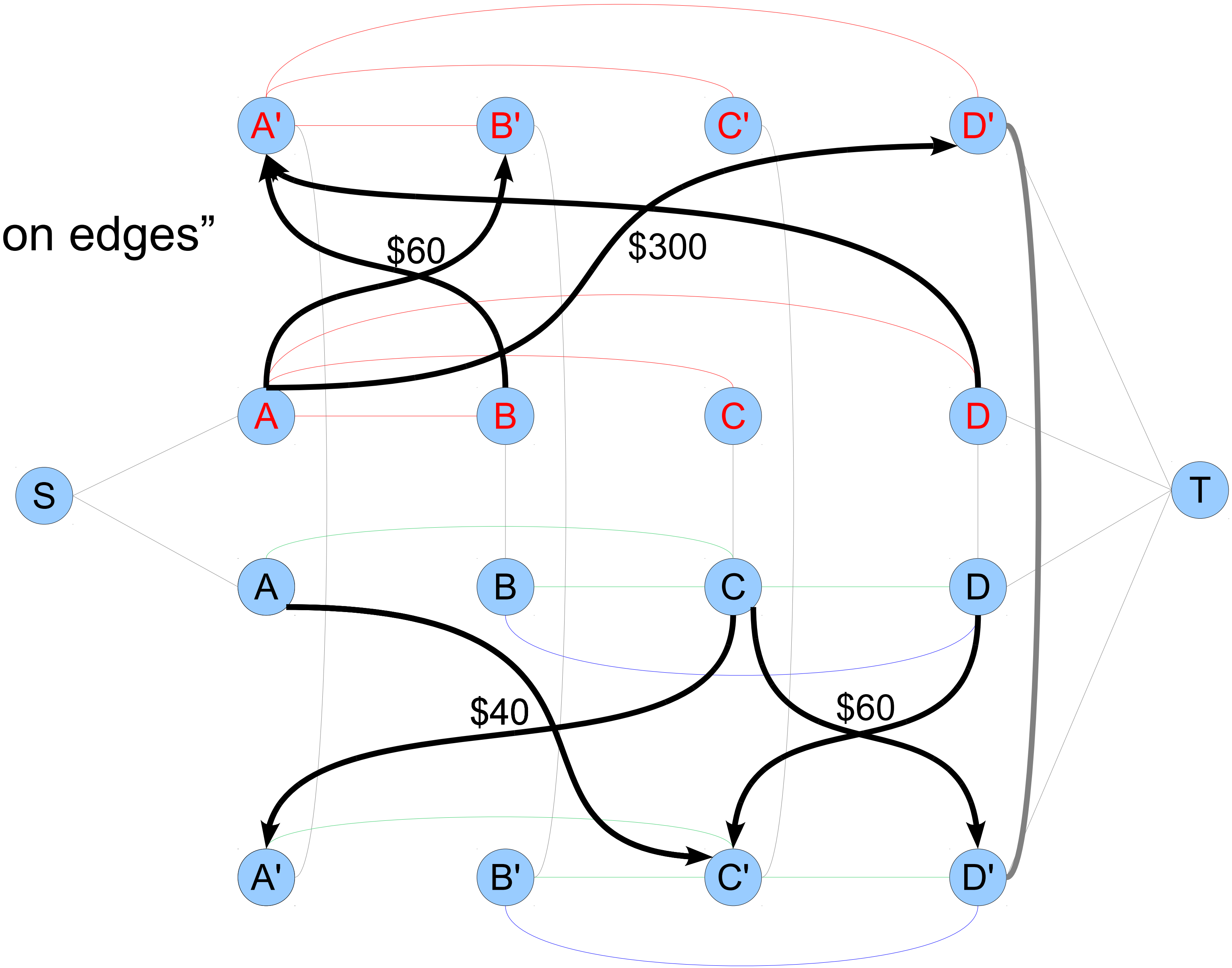
Use Coupon
on A -> B



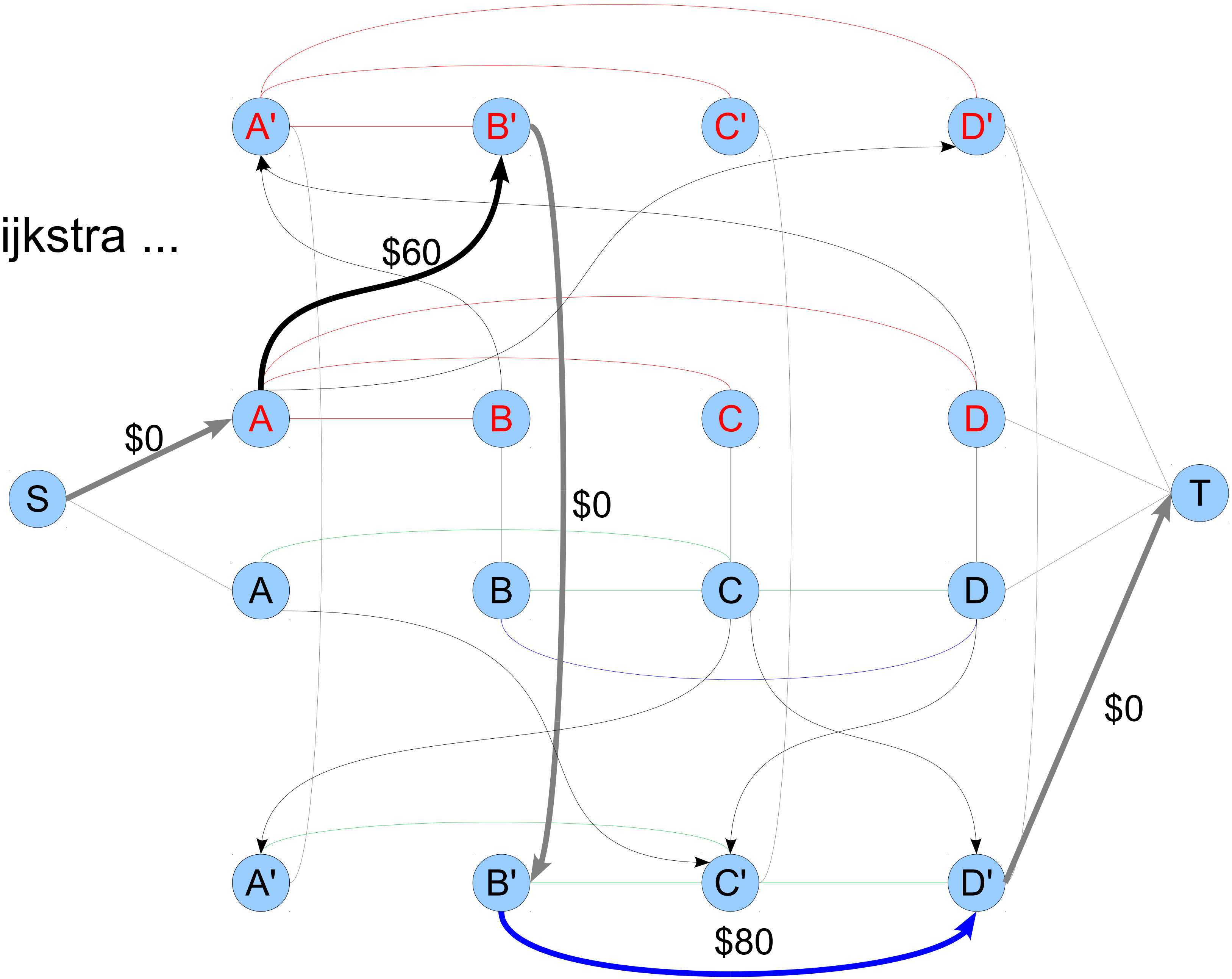
Use Coupon
on A -> C



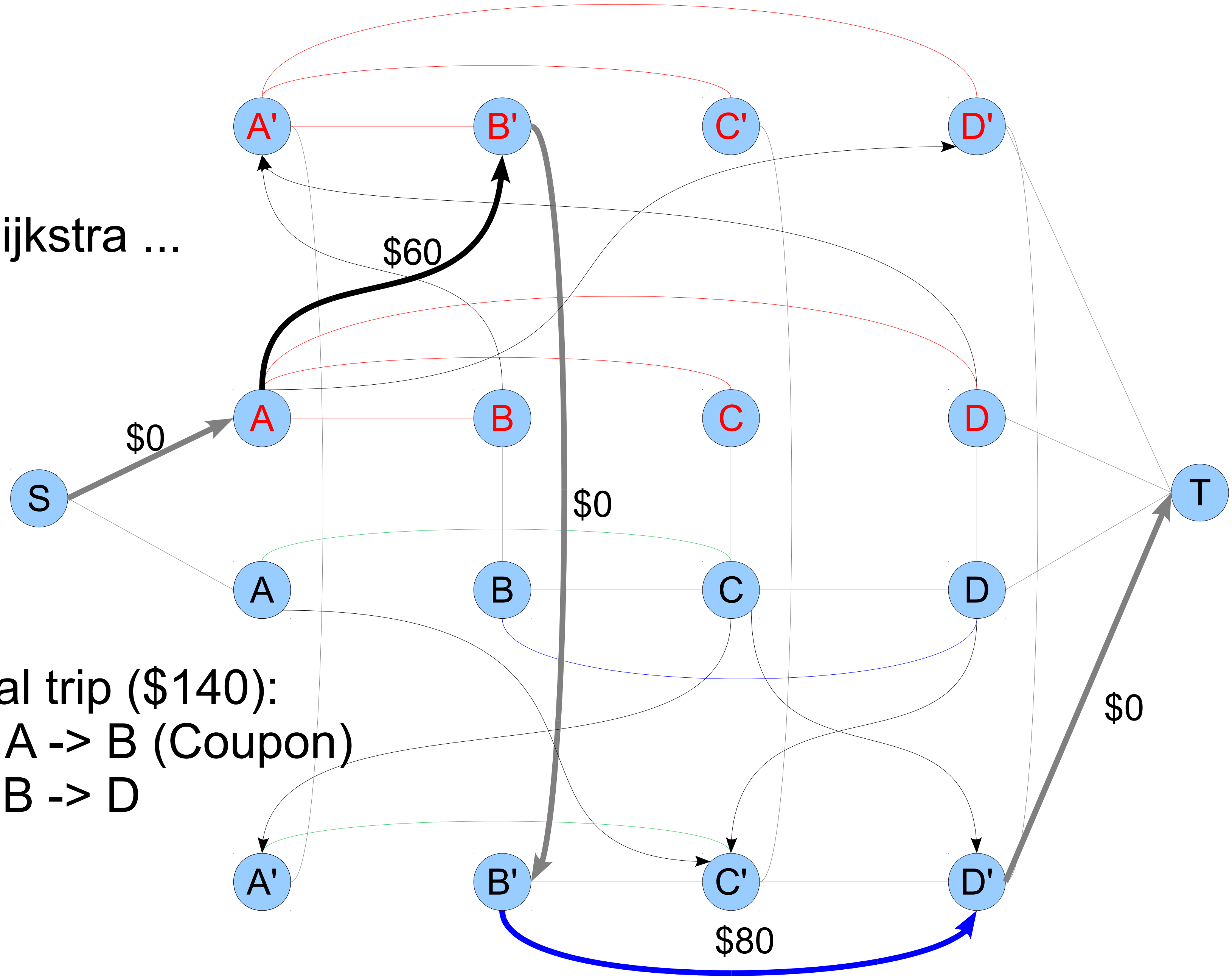
“Coupon edges”



Run Dijkstra ...



Run Dijkstra ...



Optimal trip ($140):

Red A -> B (Coupon)

Blue B -> D