Explicação do algoritmo:

O algoritmo de Johnson, assim como o de Ford Fulkerson, surgiu para resolver o problema do caminho mais curto de todos os pares em um grafo direcionado de peso esparso(com valor zero ou não presentes/necessários). Seu funcionamento se dá da seguinte forma:

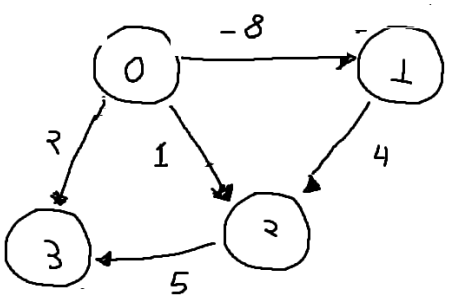
1º passo: É adicionado um novo nó com arestas de peso zero para todos os nós

2º passo: É executado o algoritmo de Bellman-Ford, verificando se há ou nao ciclos de peso menor que zero e também para encontrar X(v), que é o menor peso de um caminho do novos nós "v"

3º passo: É refeito os pesos de todas as arestas usando os valores de X(v).

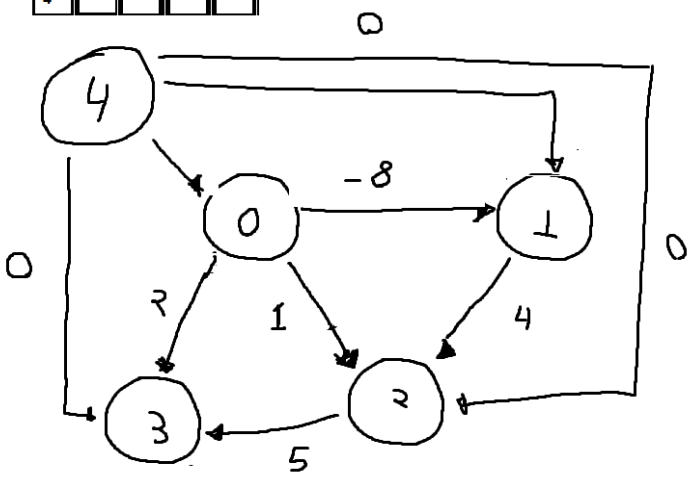
4º Finalmente, para cada nó, ele executa o algoritmo de Dijkstra e armazena o menor peso calculado para os outros nós, refazendo o peso usando os valores de X(v) dos nós como o peso final.

Funcionamento:



Dado o grafo acima, vamos aplicar o primeiro passo:

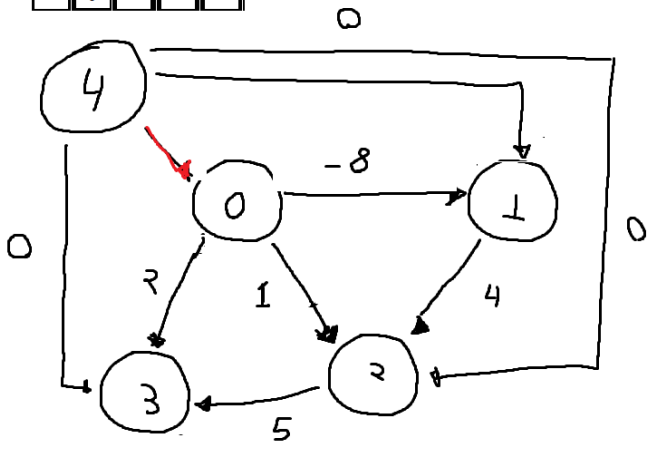
dist	0	1	2	3
4				



Agora, realizando o segundo passo:

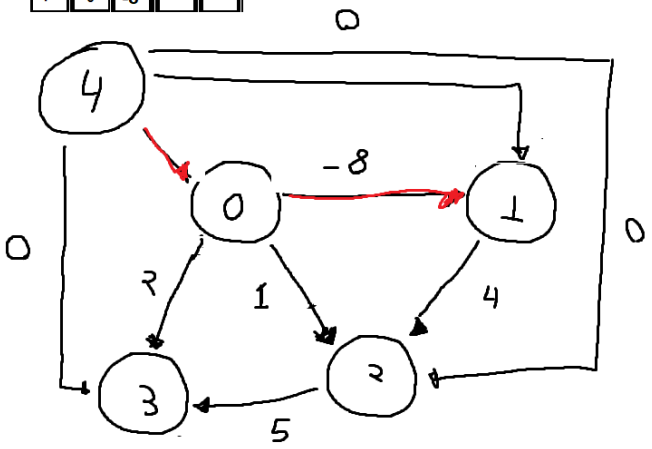
caminho

dist	0	1	2	3
4	0			



caminho

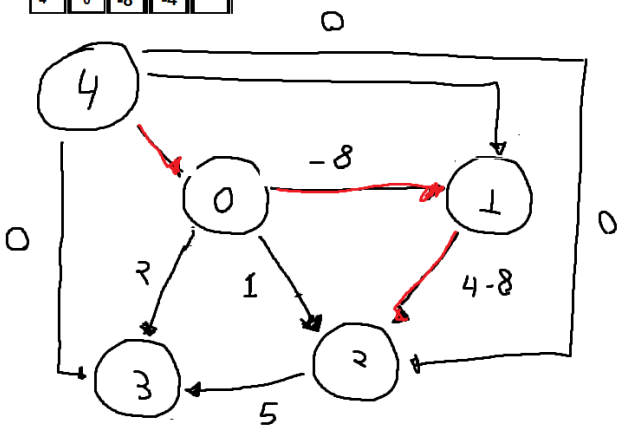
dist	0	1	2	3
4	0	-8		



caminho

dist	0	1	2	3
4	0	-8	-4	

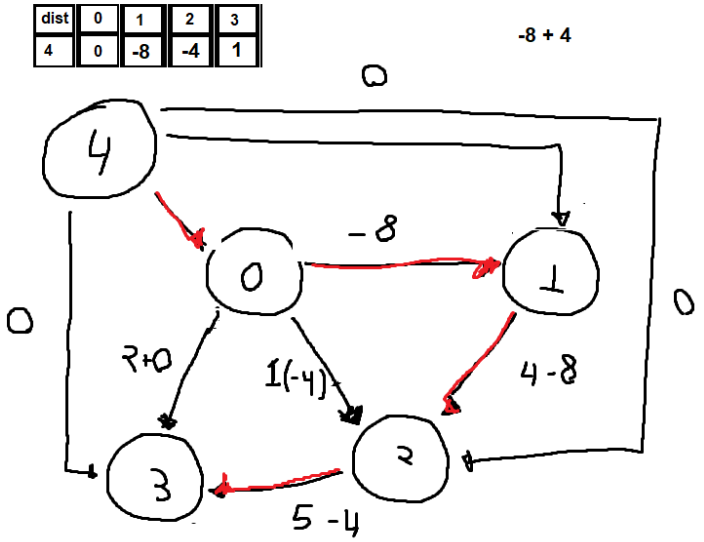
-8 + 4



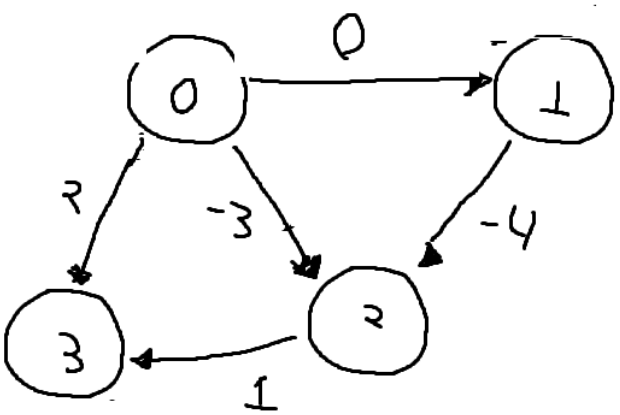
Por fim no terceiro passo, obtemos o seguinte resultado, já deixando o grafo pronto para o quarto passo:

caminho

dist	0	1	2	3
4	0	-8	-4	1



dist	0	1	2	3
4	0	-8	-4	1



Análise de complexidade:

Como na execução primeiro rodamos o Bellman-Ford (complexidade de vértices vezes arestas) e para cada vértice é chamamos o Dijkstra (complexidade log vezes vértices), fazemos (V x A) x (V log V) = Θ V^2 x log V + VA. Para um grafo completo(onde cada par de vértices distintos é conectado por uma única aresta), o custo é O(V^2) ,o mesmo Floyd Warshell, tendo que passar por todas combinações.

- referências utilizadas (1.0)

https://www.geeksforgeeks.org/johnsons-algorithm/

https://www.geeksforgeeks.org/johnsons-algorithm-for-all-pairs-shortest-paths-implementation/

https://www.youtube.com/watch?v=hLEgT-2t8Ag