Universidade de São Paulo - ICMC
SCC0216 Modelagem Computacional em Grafos

Trabalho Prático

Prof. Dr. Alneu de Andrade Lopes - 1 sem. 2013
Estagiário PAE: Alan Valejo

Junho 17, 2012

1 Trabalho

O trabalho descrito a seguir estende o algoritmo de *Dijkstra* de tal forma que essa versão possa ser empregada em um grafo cujo tamanho tornaria impraticável o uso do algoritmo básico. A ideia de tal implementação é utilizar uma heurística na seleção dos vértices de tal forma que o algoritmo busque o menor caminho entre um vértice de origem e um de destino, dados, evitando explorar caminhos pouco promissores. A seguir os detalhes da implementação e avaliação do trabalho são apresentados.

2 Implementação (50% da nota final)

A Parte II consiste na implementação em C/C++ do algoritmo de caminhos mínimos *Dijkstra* para encontrar o menor percurso entre duas cidades brasileiras. Além disso, é proposto uma variação utilizando além do custo real entre vértices, a adição do custo estimado entre um determinado vértice e o vértice de destino.

O algoritmo de *Dijkstra* considera um conjunto S de menores caminhos, iniciado com um vértice inicial I. A cada passo do algoritmo busca-se nas adjacências dos vértices pertencentes a S aquele vértice com menor distância relativa a I e adiciona-o. Repete-se os passos até que todos os vértices alcançáveis por I estejam em S.

Podemos modificar o algoritmo de *Dijkstra* utilizando uma heurística para compor a função de custo. Para encontrar o menor custo, vamos considerar uma outra medida, dada pela ccombinção da distância entre o vértice em questão n e o vértice inicial I mais a distância estimada entre este vértice n e o vértice de destino.

Em outras palavras, considera-se, além do custo real (custo acumulado de I até o vértice atual), o custo estimado do menor caminho do vértice atual até o destino final. Considerando os detalhes apresentados, pode-se definir uma função **f(n)** como: **f(n) = g(n) + h(n)**, onde:

- **g(n)** é o custo real do caminho entre até de I até n
- **h(n)** é uma estimativa do custo de n até o destino.

Assim, $f(n)$ é o custo estimado da solução de custo mais baixo passando por n. Observa-se que o algoritmo de Dijkstra utiliza apenas a função $g(n)$, mantendo na lista S sempre o menor caminho até

o vértice, n, atual. Na mudança proposta, espera-se que adicionando a estimativa do custo restante o algoritmo possa evitar caminhos pouco promissores e diminuir o tempo de busca. Uma estimativa que sempre subestima o comprimento real do caminho até o objetivo é chamada de admissível. O uso de uma estimativa admissível garante que a busca de custo uniforme ainda encontrará o menor caminho. A heurística admissível a ser utilizada é a distância em linha reta entre origem e destino (Ver Tabela 3).

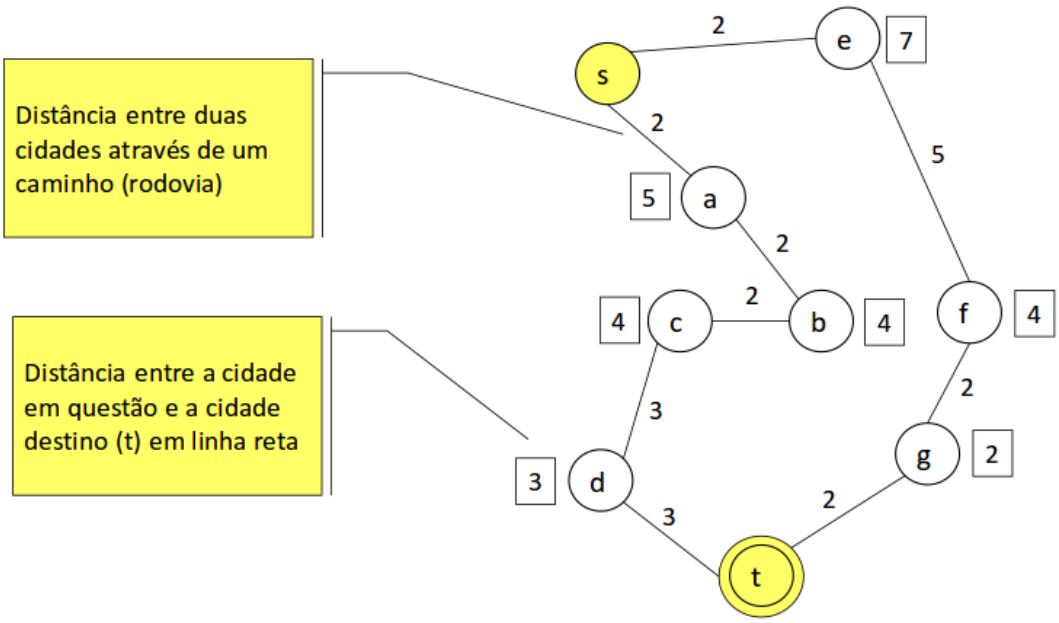

Figura 1: Ilustração da abordagem proposta.

Por exemplo, a Figura 2 demostra a execução do algoritmo de caminhos mínimos utilizando, além do custo real do caminho entre dois vértices, a distância em linha reta entre um determinado vértice e o destino.

O algoritmo inicia a execução pelo vértice fonte s, e quer encontrar o menor caminho até t. Uma primeira iteração calcula o custo estimado do vértice s até t, passando por a e e. Como o custo estimado do caminho que passa por a é menor que o custo estimado do caminho que passa por e, o algoritmo prefere seguir por a ao invés de e. Ao termino de execução do algoritmo, o caminho mínimo entre s e t é destacado por e, f e g, com custo final de 11.

A partir do formalismo do algoritmo de *Dijkstra*, implemente as aalterações necessárias, a fim de ajustar o algoritmo, considerando além do custo real entre dois vértices o custo estimado entre o vértice em questão e o vértice final.

O programa deve ser testado sobre o mapa rodoviário das capitais dos Estados do Brasil apresentado na Figura 3. Dado uma cidade de origem e outra de destino, deve-se apresentar um percurso, dado por uma lista de cidades, que representa a trajetória de menor custo. Sua implementação deve utilizar *flags* que permitam ligar/desligar cada um dos algoritmos em questão.

Os nomes das 27 capitais dos Estados são apresentados na Tabela 1. As distâncias rodoviárias (em Km.) entre as capitais são apresentadas na Tabela 2. Na Tabela 3 apresentam-se as distâncias (em Km.) em linha reta entre as capitais (aproximadas via distância aérea entre as mesmas).

- f(a)=g(a)+dist(a,t)=2+5=7
- f(e)=g(e)+dist(e,t)=2+7=9
- f(b)=g(b)+dist(b,t)=4+4=8
- f(c)=g(c)+dist(c,t)=6+4=10
- f(f)=g(f)+dist(f,t)=7+4=11
- f(d)=g(d)+dist(d,t)=9+3=12
- f(g)=g(g)+dist(g,t)=9+2=11
- f(t)=g(t)+dist(t,t)=11+0=11

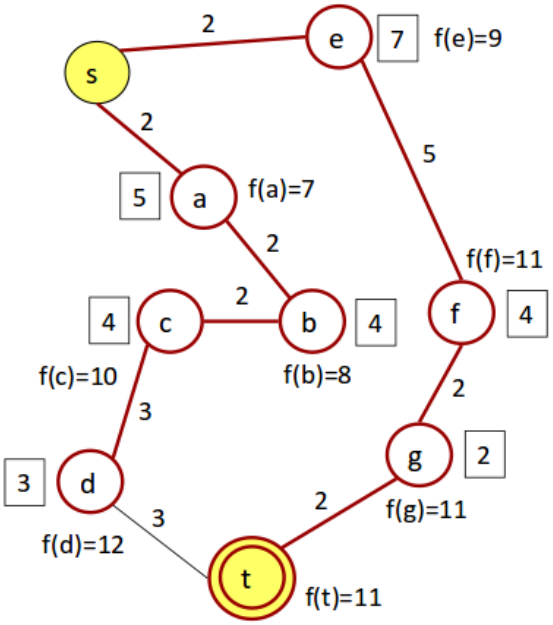

Figura 2: Exemplo de execução da abordagem proposta. Nesse, o caminho mínimo entre s e t é destacado por e, f e g, com custo final de 11.

3 TAD (30% da nota final)

É importante considerar os algoritmos em grafos como Tipos Abstratos de Dados, criando um conjunto de operações associadas a uma estrutura de dados, a fim de garantir a independência de implementação para as operações. Discutimos em aula uma estrutura genérica para grafos, capaz de armazenar qualquer tipo de informação. Estenda o TAD Grafos visto em aula, a fim de adequar a modelagem ao mapa rodoviário das capitais dos Estados do Brasil, apresentado na Figura 3.

Observação: Organize a sua implementação. Defina um Tipo Abstrato de Dados (TAD) para as estruturas de dados auxiliares. Por exemplo, em C crie módulos para as estruturas de dados e defina uma interface para cada estrutura de dados com as operações definidas pelo TAD em um arquivo *header* (.h). Separe a definição da interface da implementação das operações criando um arquivo fonte (.c) para cada arquivo *header*. Em uma linguagem de programação orientada a objetos como C++, crie uma classe para cada estrutura de dados e defina a interface da classe com as operações do TAD.

4 Avaliação de Desempenho (20% da nota final)

A avaliação de desempenho consiste em comparar os tempos de execução do algoritmo de *Dijkstra* com o uso da heurística proposta. Como medir o tempo em termos de tempo de execução utilizando um relógio pode levar a resultados que variam com a linguagem de programação, computador e outros fatores, neste projeto o desempenho será avaliado pelo número de vértices visitados durante a busca até se encontrar uma primeira solução. Para aperfeiçoar os resultados da avaliação de desempenho deve-se criar novas composições da função h(n). Por exemplo, considerar o dobro e o triplo da distância em linha reta. Faça um relatório (1 ou 2 páginas) explicando sua implementação e os resultados obtidos na parte 2.

Tabela 1: Nomes das capitais dos 27 Estados do Brasil.

Id	Nome
1	Boa Vista
2	Macapá
3	Manaus
4	Belém
5	São Luis
6	Teresina
7	Fortaleza
8	Natal
9	João Pessoa
10	Recife
11	Maceió
12	Aracaju
13	Salvador
14	Rio Branco
15	Porto Velho
16	Cuiabá
17	Palmas
18	Goiânia
19	Brasília
20	Belo Horizonte
21	Vitória
22	Rio de Janeiro
23	Campo Grande
24	São Paulo
25	Curitiba
26	Florianópolis
27	Porto Alegre

Tabela 2: Distância rodoviária em Km que representam as conexões entres cidades na Figura 1.

Origem	Destino	Distância
1	3	785
3	14	1445
3	15	901
3	4	5298
3	17	4141
4	17	1283
4	5	806
5	17	1386
6	17	1401
6	7	634
6	8	1171
6	19	1789
6	9	1224
9	10	120
9	7	688
10	11	285
10	19	2135
11	12	294
11	19	1930
12	13	256
13	19	1446
13	20	1372
14	15	544
14	16	1990
15	16	1456
15	18	2390
16	23	694
17	18	874
17	19	973
18	19	209
18	20	906
18	23	935
19	20	716
20	21	524
20	23	1453
20	24	586
21	24	882
21	13	1202
21	22	521
22	24	429
22	26	1144
22	25	852
22	20	434
23	24	1014
23	25	991
24	25	408
25	26	300
25	27	711
26	27	476

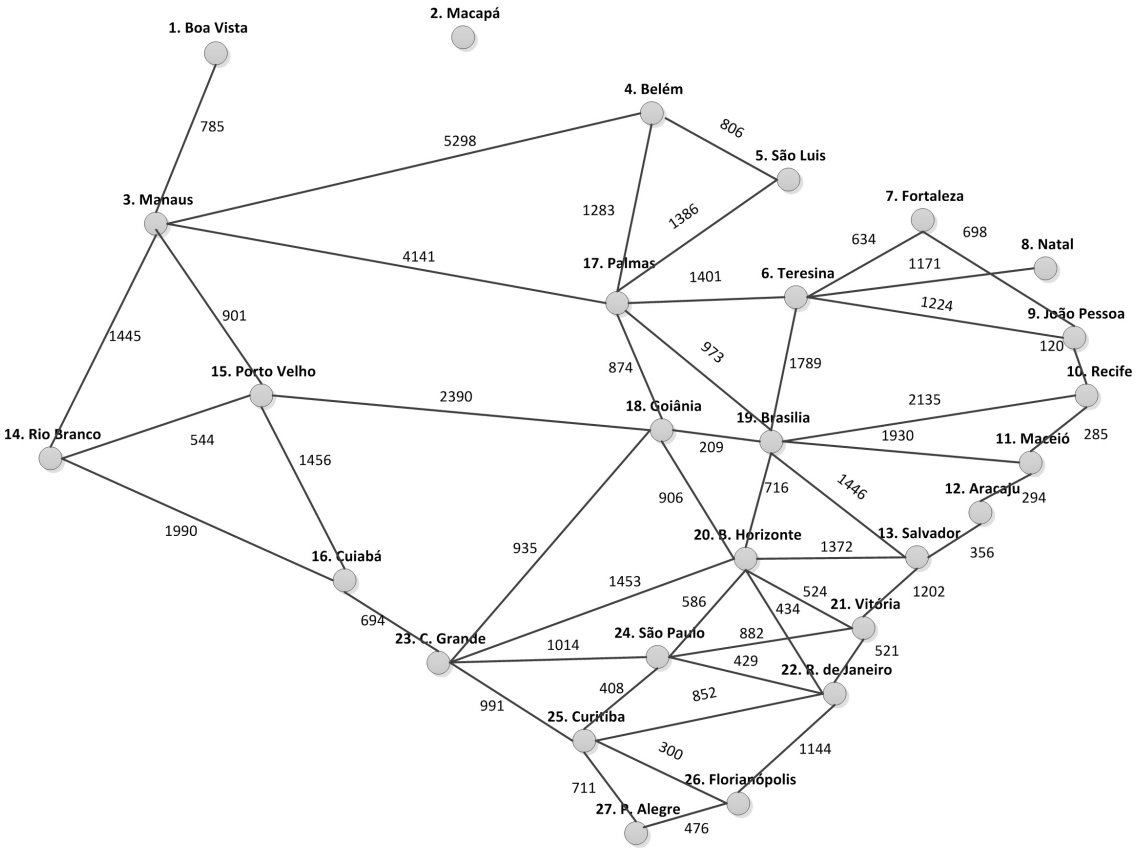

Figura 3: Mapa rodoviário das capitais dos Estados do Brasil

Tabela 3: Distância rodoviária em Km que representam as conexões entres cidades na Figura 1.

	1	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
1	0	661	1432	1913	2169	2562	2983	3067	3103	3089	3022	3009	1626	1335	2107	1988	2503	2496	3117	3394	3428	2667	3300	3370	3620	3785
3		0	1292	1746	1921	2383	2765	2819	2833	2778	2673	2605	1149	761	1453	1509	1912	1932	2556	2865	2849	2013	2689	2734	2981	3132
4			0	481	750	1133	1550	1636	1676	1680	1641	1687	2333	1886	1778	973	1693	1592	2111	2275	2450	2212	2463	2665	2904	3188
5				0	2091	652	1071	1162	1209	1234	1226	1323	2726	2274	1942	964	1662	1524	1932	2023	2266	2284	2348	2599	2821	3142
6					0	495	843	905	934	929	903	994	2806	2362	1862	835	1467	1313	1652	1713	1979	2132	2091	2362	2573	2909
7						0	435	555	629	730	815	1028	3300	3213	2329	1300	1854	1687	1893	1855	2190	2547	2368	2670	2857	3213
8							0	151	253	434	604	875	3616	3179	2524	1527	1948	1775	1831	1706	2085	2654	2320	2645	2802	3172
9								0	104	299	486	763	3632	3200	2495	1521	1889	1716	1726	1581	1968	2593	2216	2545	2693	3066
10									0	202	398	675	3618	3190	2452	1498	1829	1657	1639	1483	1874	2530	2128	2459	2603	2977
11										0	201	475	3510	3090	2302	1383	1656	1485	1439	1282	1671	2352	1928	2259	2402	2775
12											0	277	3359	2946	2121	1235	1461	1292	1248	1102	1482	2155	1731	2061	2207	2580
13												0	3206	2808	1915	1114	1225	1060	964	839	1209	1905	1453	1784	1930	2303
14													0	449	1414	2127	2138	2246	2786	3156	2982	1827	2704	2601	2809	2814
15														0	1137	1711	1813	1900	2477	2835	2707	1634	2463	2412	2641	2706
16															0	1029	740	873	1372	1745	1575	559	1326	1302	1543	1679
17																0	724	620	1178	1413	1512	1320	1493	1693	1931	2222
18																	0	173	666	1022	936	705	810	972	1215	1497
19																		0	624	947	933	878	873	1081	1314	1619
20																			0	378	339	118	489	820	973	1341
21																				0	412	1490	741	1076	1160	1536
22																					0	1212	357	675	748	1123
23																						0	894	780	1007	1119
24																							0	338	489	852
25																								0	251	546
26																									0	376
27																										0