Optimization and Algorithms

Reserving optimal link capacity in a network

Marco Graca João Loureiro Pedro Vasco
marco.graca@tecnico.ulisboa.pt joao.firmino@tecnico.ulisboa.pt
pedro.vasco@tecnico.ulisboa.pt
Instituto Superior Técnico

***Resumo*—O problema consiste em determinar a quantidade que seria necessário reservar numa rede composta por nós e arestas que os ligam, de modo a garantir o correto transporte de todos os 'flows' em diferentes cenários com um custo mínimo respeitando o sentido e capacidade de cada aresta. Este problema reduz-se a um problema linear com um número de variáveis e restrições da ordem do número de arestas vezes o número de cenários.**

I. INTRODUÇÃO

O problema consiste em determinar a reserva r mínima para cada aresta de uma rede, de modo a satisfazer os possíveis flows f. É dada uma descrição da rede, o conjunto de cenários de tráfego e a capacidade e função de custo de cada aresta da rede.

Este tipo de problema é comum nas mais variadas áreas. Neste caso falamos de capacidades numa rede de internet mas o problema pode ser aplicado a qualquer rede de distribuição. Por exemplo, imagine-se um dado volume de encomendas recebidas em vários pontos que têm que ser distribuídas em outros pontos. Considerando as arestas como estradas em que o seu custo é a distância ou velocidade a que se transita nelas, queremos minimizar a distância percorrida ou o tempo gasto.

Este problema pode ser descrito como um problema linear. Para problemas de tamanho moderado pode ser resolvido com algoritmos lineares, no entanto, para problemas de maior escala, estes algoritmos não conseguem encontrar soluções em tempo razoável.

Temos quatro vertentes diferentes do problema, posto isto decidimos dividir o relátorio em quatro partes abordando cada uma delas individualmente. Cada uma destas partes está por sua vez organizada em 4 partes. Na introdução é feita uma descrição sucinta do problema que queremos abordar. Na formulação do problema são explicadas as variáveis usadas mencionando quais delas são variáveis a otimizar. Na abordagem é descrito qual o método usado para resolução do problema formulado. Nos resultados numéricos são analisados os resultados obtidos fazendo um paralelo com a solução esperada.

II. PARTE 1

A. Introdução

Esta parte do problema corresponde à sua versão mais simples, caso em que o custo de cada aresta por unidade reservada p é 1, ou seja há uma proporcionalidade direta. Considera-se que a capacidade máxima c de cada aresta é 1.

B. Formulação do Problema

O problema descrito é formulado matematicamente da seguinte maneira:

$$\begin{array}{ll} \underset{f,r}{\text{minimize}} & p \cdot r \\ \text{subject to} & Af_i + s_i \leq 10^{-10} \\ & 0 \leq f_i \leq r \\ & r \leq c \end{array} \tag{1}$$

A matriz A, em que cada coluna representa uma aresta e cada linha um nó, define a orientação das mesmas da seguinte forma:

$$A = \begin{cases} 1 & \text{se a aresta entra no nó,} \\ -1 & \text{se a aresta sai do nó,} \\ 0 & \text{caso contrário} \end{cases} \tag{2}$$

C. Abordagem

Para resolver este problema usámos o algoritmo cvx que é usado para resolver problemas convexos. CVX transforma o MATLAB numa linguagem de modelação permitindo especificar restrições e objetivos usando a sintaxe standard do MATLAB. Funções connvexas são construídas através de um pequeno conjunto de regras de análise convexa. Restrições e objetivos construídos usando estas regras são automaticamente transformadas numa forma canónica e resolvidas.[1]

Como o nosso problema é convexo, pois trata-se de uma função afim, podemos usar este algoritmo. De notar, que o algoritmo CVX não é adequado para problemas com muitas variáveis logo caso tenhamos um número excessivamente elevado de nós ou arestas teremos que escolher outra abordagem.

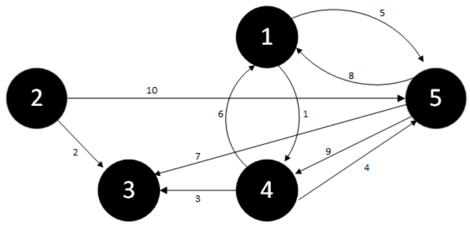

Figura 1: Grafo representativo da rede

D. *Resultados Numéricos*

Para testar a nossa formulação e abordagem ao problema foi usada a rede da figura 1.

Os cenários (o que se pretende transportar entre nós) estão representados na seguinte matriz s (N*M) sendo N o número de nós e M o número de diferentes cenários.

$$s = \begin{bmatrix} 0 & 0.5 & 0 & -1 & 0.75 & 0 & -1.75 & 0.25 \\ 0.25 & 1.5 & 0.75 & 0.75 & 1.25 & 1.25 & 0.75 & 1.5 \\ -1 & -1 & -0.5 & -1.5 & -1.5 & -2.25 & -0.75 & -1.5 \\ 2 & 1 & -0.5 & 0 & 0 & 2 & 0.25 & 1.25 \\ -1.25 & -2 & 0.25 & 1.75 & -0.5 & -1 & 1.5 & -1.5 \end{bmatrix}$$

Tendo em conta os parâmetros iniciais anteriores, o algoritmo descrito conduz á seguinte solução de reserva para cada aresta:

$$r = \begin{bmatrix} 0 \\ 0.8732 \\ 0.8329 \\ 0.5352 \\ 0.7981 \\ 0.75 \\ 0.5439 \\ 1.0 \\ 0.5 \\ 0.666 \end{bmatrix}$$

Sendo a função de custo total igual a 6.5. Tendo em conta estes resultados podemos concluir que, dado o custo ser proporcional, o algoritmo procura distribuir os fluxos usando o menor número de arestas possível. O que se pretendia minimizar era a variável de custo total que para este caso é igual a 6,5.

III. Parte 2

A. *Introdução*

Nesta segunda parte foi introduzida uma alteração ao custo de algumas arestas que passaram a ter um *overhead*, ou seja, um custo fixo por estarem a ser usadas. Todos os outros parâmetros do problema mantêm-se como anteriormente.

B. *Formulação do Problema*

De modo a ter em conta a particularidade descrita anteriormente foi criado um vetor E contendo o valor de *overhead* de cada uma das arestas (arestas sem *overhead* têm valor 0) e um vetor g que terá valor 0 ou 1 consoante a aresta esteja ou não a ser usada, de maneira a não contabilizar o *overhead* de arestas que não são usadas. O problema será então formulado da seguinte maneira:

$$\begin{array}{ll} \underset{f,r,g}{\text{minimize}} & p \cdot r + g \cdot E \\ \text{subject to} & Af_i + s_i \leq 10^{-10} \\ & 0 \leq f_i \leq r \\ & r \leq g \cdot c \\ & 0 \leq g \leq 1 \end{array} \tag{3}$$

Nesta formulação, definimos que g pode tomar qualquer valor entre zero e um de modo a manter o problema linear. Isto não é verdade, g pode apenas ser 0 ou 1 pois uma aresta está ou não a ser usada, não há meio termo. Isto implica que terá que ser feito um pós-processamento de g, de maneira a fixar os seus valores em 0 ou 1.

C. *Abordagem*

Na abordagem a este problema usámos o algoritmo CVX descrito na parte anterior.

Uma maneira de garantir que g tomava sempre valor 0 ou 1 seria, não o usando como variável de optimização, correr o algoritmo com as várias combinações possíveis de g para as arestas com *overhead* e escolher a combinação que gera um custo total menor. Porém, este método de "força bruta" implica um grande número de iterações que pode vir a consumir muito tempo e capacidade computacional ao aumentar o tamanho do problema.

A alternativa é a descrita anteriormente na formulação do problema onde o pós-processamento de g corresponde à definição de um valor de *threshold* que separa valores que devem ser 0 ou 1 ou a uma versão do método de força bruta, onde se testam as combinações apenas dos valores que não foram já definidos como 0 ou 1 das arestas com *overhead*. De notar que a primeira opção, apesar de ser mais simples, pode resultar em valores inválidos dado que pode ser obtido um valor de g abaixo do *threshold* definido numa aresta necessária para satisfazer todos os fluxos.

D. *Resultados Numéricos*

Para testar a nossa formulação usámos a rede e os cenários referidos na parte anterior. Foram adicionados *overheads* de valor 1 nas arestas 1, 6 e 7 enquanto que as restantes mantiveram-se com custo proporcional.

Tendo em conta os parâmetros iniciais anteriores, o algoritmo descrito conduz á seguinte solução onde se apresentão a negrito os valores das arestas com *overhead*:

$$g = \begin{bmatrix} \mathbf{0} \\ 1.0 \\ 1.0 \\ 0.824 \\ 0.908 \\ \mathbf{0.75} \\ \mathbf{0.25} \\ 1.0 \\ 0.798 \\ 0.837 \end{bmatrix}, r = \begin{bmatrix} 0 \\ 1.0 \\ 1.0 \\ 0.581 \\ 0.799 \\ 0.75 \\ 0.25 \\ 1.0 \\ 0.5 \\ 0.620 \end{bmatrix}$$

Verificamos, como previsto, que o vetor g não tem apenas valores 0 ou 1. Como não existe hipótese de haver um valor entre usar ou não a aresta temos que ter aqui um critério de decisão. Note-se que, no vetor g, apenas os valores das arestas com *overhead* são relevantes.

Em suma, usando a segunda alternativa (definiçao de um *threshold*) com um valor de 0.1 obtemos a seguinte solução nas reservas da rede anterior:

$$g = \begin{bmatrix} \mathbf{0} \\ 1.0 \\ 1.0 \\ 1.0 \\ 1.0 \\ \mathbf{1.0} \\ \mathbf{1.0} \\ 1.0 \\ 1.0 \\ 1.0 \end{bmatrix}, r = \begin{bmatrix} 0 \\ 0.8761 \\ 0.8335 \\ 0.5388 \\ 0.7975 \\ 0.75 \\ 0.5403 \\ 1.0 \\ 0.5 \\ 0.6637 \end{bmatrix}$$

Estes resultados foram confirmados com o método ”força bruta” onde se verificou que, para os cenários considerados nesta rede, as arestas 6 e 7 eram essenciais para satisfazer todos os fluxos. O custo obtido para esta solução foi de 8.5, 2 a mais que anteriormente, devido às duas arestas com um overhead de valor 1 a serem utilizadas, dado que a quantidade reservada é a mesma. Para testar o funcionamento do algoritmo foi feita a experiência de aumentar significativamente o *overhead* de uma das arestas e verificou-se que essa aresta deixou de ser reservada.

IV. Parte 3

A. Introdução

Neste parte do problema é abordada uma possível situação em que o sentido de algumas arestas não são conhecidas a priori. Ou seja, terá que ser feita a reserva de arestas tendo em conta que a rede poderá ser uma de duas redes possíveis que diferem no sentido de algumas arestas. Nesta parte os custos das arestas serão todos lineares, sem *overheads*.

B. Formulação do Problema

Esta vertente do problema pode ser descrita da seguinte forma:

$$\begin{array}{ll} \underset{f,h,r}{\text{minimize}} & p \cdot r \\ \text{subject to} & 0 \leq f_i <= r \\ & 0 \leq h_i <= r \\ & r \leq c \\ & A \cdot f_i + s_i \leq 10^{-10} \\ & As \cdot h_i + s_i \leq 10^{-10} \end{array} \quad (4)$$

As variáveis f e h que queremos determinar representam os flows nas duas redes diferentes. A matriz A representa a rede original enquanto que As representa a mesma matriz A no entanto multiplicada por uma matriz diagonal em que os elementos -1 dessa matriz diagonal representam as arestas que mudam de direção. Em suma, a matriz As é a matriz A com as arestas que mudam no sentido oposto.

C. Abordagem

Na abordagem a este problema usámos o algoritmo CVX descrito na parte anterior.

Esta abordagem é semelhante às anteriores mas, em vez de minimizar a reserva de uma só rede, minimiza uma reserva que se adapte às duas redes. Isto permite prevenir eventuais mudanças futuras na rede pois com isto, mesmo com mudanças, a solução determinada mantem-se adequada.

D. Resultados Numéricos

Para testar a formulação considerámos a rede da figura 2 em que as arestas ponteadas são as que são passivas de mudar de direção.

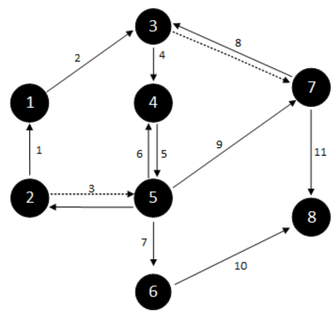

Figura 2: Grafo representativo da rede

Num cenário é pretentido enviar uma unidade da aresta 2 para a 7 e no outro cenário outra unidade da aresta 2 para a 6. Os resultados do vetor r são os seguintes:

$$\begin{bmatrix} 1.0 & 1.0 & 0 & 1.0 & 1.0 & 0 & 1.0 & 0 & 1.0 & 0 & 0 \end{bmatrix}$$

Pela análise dos resultados, podemos que verificar que este tipo de formulação, ao não ter garantias de poder usar uma dada aresta irá procurar um caminho alternativo reservando assim outras arestas. Isto conduzirá a um caminho mais caro mas garante que existe sempre solução. Uma análise mais superficial, daria a entender que o algoritmo simplesmente evita arestas que possam mudar de direção. Para testar essa nuance, criámos a rede da figura 3.

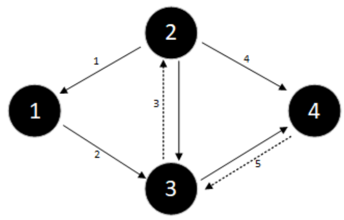

Figura 3: Grafo representativo da segunda rede

Queremos levar uma unidade de 1 para 4. Neste caso o algoritmo formulado irá reserva os dois caminhos possíveis (ou seja pela aresta 3 e 4 e pela aresta 5). Com estes resultados provamos que a nossa formulação não ignora simplesmente as arestas que mudam de direção mas sim garante que terá sempre um caminho para transportar os flows independentemente das mudanças na rede.

V. Parte 4

A. Introdução

Por fim consideramos uma situação em que o custo de cada aresta deixa de ser uma reta de declive 1 proporcional à reserva mas sim declives diferentes em diferentes intervalos de r (níveis) como se mostra na figura 4:

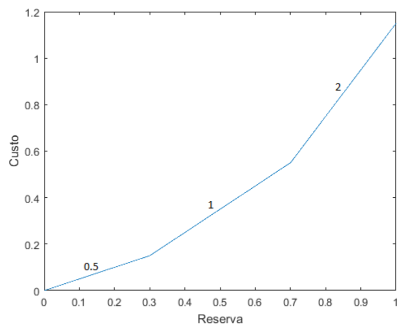

Figura 4: Gráfico representativo dos custos

B. Formulação do Problema

O problema pode ser formulado da seguinte forma:

$$\begin{array}{ll} \underset{f,r}{\text{minimize}} & l_{i2} \cdot \sum_i r_{ik} \\ \text{subject to} & 0 \leq f_{ijk} \leq r_{ik} \leq l_{k1}, \qquad \forall i,j,k \\ & A\sum_k f_{ijk} + s_j \leq 10^{-10}, \quad \forall i,j \end{array} \tag{5}$$

O vetor l tem dimensão 3×2 em que na segunda coluna temos os diferentes declives e na primeira a dimensão (capacidade) de cada intervalo com o declive correspondente. A variavel f é agora uma matriz cúbica que em cada elemento terá a distribuição do fluxo em cada aresta num dado cenário pelos três níveis de custo possíveis. A matriz r terá o valor de reserva para cada um dos níveis. De notar que devido aos declives serem sempre crescentes, o algoritmo só começa a reservar um certo nível após completar o anterior.

C. Abordagem

Na abordagem a este problema usámos o algoritmo CVX descrito na parte anterior.

D. Resultados numéricos

Foi usada a mesma rede e os mesmos cenários da parte 1. Os resultados obtidos foram os seguintes:

$$r = \begin{bmatrix} 0.050 & 0 & 0 \\ 0.300 & 0.400 & 0.120 \\ 0.300 & 0.400 & 0.019 \\ 0.300 & 0.311 & 0 \\ 0.300 & 0.400 & 0 \\ 0.300 & 0.400 & 0.050 \\ 0.300 & 0.400 & 0.011 \\ 0.300 & 0.400 & 0.300 \\ 0.300 & 0.200 & 0 \\ 0.300 & 0.389 & 0 \end{bmatrix} = \begin{bmatrix} 0.050 \\ 0.820 \\ 0.719 \\ 0.611 \\ 0.700 \\ 0.750 \\ 0.711 \\ 1.0 \\ 0.500 \\ 0.689 \end{bmatrix} \tag{6}$$

Comparando estes resultados com os da primeira parte (caso de custo proporcional com declive 1) podemos verificar que o algoritmo procura distribuir os fluxos de forma uniforme entre as arestas. Isto era o que seria de esperar teoricamente pois torna-se mais caro usar muito a mesma aresta dado o declive do custo aumentar com a quantidade que se reserva.

VI. Conclusões

Neste relatório foram apresentadas quatro variantes diferentes contendo o mesmo tipo de problema como base. Optámos por resolver os diferentes problemas fazendo uma formulação específica para cada um deles. No entanto, poderiamos agora facilmente juntar as várias formulações de modo a obter uma formulação que permitisse ter em conta todas as nuances consideradas ao mesmo tempo. Num caso da vida real onde este problema seja apresentado uma abordagem adequada seria formular algo preparado para as várias variâncias do problema base. A título de exemplo, caso não houvesse overheads, bastaria colocar o vetor E com zeros mantendo o problema funcional. O mesmo aconteceria para as outras nuances caso as mesmas não se verificassem.

Referências

[1] Michael Grant, Stephen Boyd *CVX Introduction*. available at `http://cvxr.com/cvx/` 2012