Universidade do Minho
Escola de Engenharia

Circuitos de Recolha de Resíduos Urbanos

Mestrado Integrado em Engenharia Informática

Sistemas de Representação de Conhecimento e Raciocínio

Autor:
A89616 - Eduardo Benjamim Lopes Coelho

7 de junho de 2021

Resumo

Neste projeto pretende-se implementar estratégias de procura (informada e não informada) com o objetivo de otimizar os circuitos de recolha de resíduos em Lisboa. Esta implementação será seguida de uma análise crítica dos resultados da aplicação dos diferentes algoritmos ao grafo gerado pelos dados da distribuição de pontos de Recolha na freguesia de Misericórdia.

Índice

0.1 Introdução

O objetivo deste projeto é desenvolver um sistema que permita importar os dados relativos aos diferentes circuitos de recolha de resíduos e representá-los numa base de conhecimento e que seja capaz de recomendar circuitos de recolha para o caso de estudo.

O circuito está dividido nas seguintes considerações elementares:

- Inicial – Percurso entre a garagem e o primeiro ponto de recolha;
- Ponto de Recolha - Remoção de resíduos num local de paragem;
- Entre Pontos – Percurso entre dois pontos de Recolha;
- Transporte – Percurso entre o último ponto de Recolha e o local de deposição de resíduos;
- Final – Percurso entre o local de deposição e a garagem.

O circuito começa com a saída do veículo da garagem, durante o circuito é efetuada uma paragem em cada ponto de recolha para remover os resíduos desse ponto. Neste projeto estamos a considerar uma versão simplificada do problema em que não é tida em conta a carga máxima do camião. O percurso termina quando o camião se dirige a um local próprio para descarregar e regressa à garagem descarregada.

Formulação do problema

1.1 Tipo de Problema

Estamos perante um problema de múltiplos estados, uma vez que o ínicio e fim do percurso (garagem) pode ser definido em qualquer ponto. É um ambiente determinístico e totalmente observável

1.2 Estados

Os estados neste problema são os diferentes nodos/pontos de recolha em que o camião pode estar. Como, neste problema, um circuito começa e acaba na garagem do camião, então o nodo onde a garagem está definida é, simultâneamente, estado inicial e final.

1.3 Operadores

Os operadores deste problema são as transições entre pontos de recolha, através de arestas que têm um determinado custo. Devido à enorme quantidade de transições possíveis, estão presentes em anexo nos apêndices

1.4 Teste Objetivo

Para sabermos que já chegámos à solução e que estamos no fim do problema, então é necessário definir um teste objetivo. Neste caso, sabemos que o circuito está terminado quando o estado atual é a garagem. Porém, como a garagem é, simultaneamente, o estado inicial e final, é preciso ser satisfeita outra condição: o camião tem de ter descarregado os resíduos no local de depósito.

1.5 Custo da Solução

O Custo da solução corresponde ao somatório dos custos das arestas que fazem parte do circuito final. O custo de uma aresta é definido pela distância entre as coordenadas dos pontos de Recolha em questão.

Processamento dos Dataset

O Dataset fornecido não está em condições de ser utilizado diretamente pelo sistema. Assim, foi necessário criar um parser, acompanhado de uma gramática tradutora para processar a informação e exportar a mesma num formato mais adequado. Para tal, foram utilizados os módulos PLY e YACC do Python (Foi também utilizado o

módulo geopy para calcular distâncias entre coordenadas). A interpretação dos dados seguida neste projeto foi a seguinte:

Utilizando a seguinte entrada como exemplo:

$-9.14330880938.70807879355 Misericordia 15805 : RdoAlecrim(Par(\rightarrow)(26 \rightarrow 30) : RFerragial - RAtaide) LixosCV 009090190$

Os dois primeiros valores correspondem à latitude e longitude do ponto de recolha em questão, respetivamente. Seguidamente, temos o número da entrada do Dataset, seguido da Freguesia. De seguida, e talvez, o campo mais importante temos a informação relativa às adjacências entre ruas/pontos de recolha. É neste campo de informação onde a maior parte dos problemas surgiram, uma vez que existem imensas entradas com formatos diferentes. O modo como está ser interpretado está detalhadamente explicado posteriormente. Por último, temos os campos relativos às informações do contentor de resíduos em questão: tipo de resíduo, tipo de contentor, capacidade e quantidade de contentores e por último a capacidade total que resulta do produto dos dos dois valores anteriores.

O campo de informação relativo às adjacências entre ruas e pontos de recolha inicia sempre com o ID do ponto de recolha em questão (identificador único do ponto de recolha). Esta valor é sempre seguido do nome da rua atual. A seguir é onde os problemas e ambiguidades surgem. Por isso, com recurso aos próximos exemplos, irei demonstrar a interpretação que dei a cada um dos formatos:

15805: R do Alecrim (Par (-¿)(26-¿30): R Ferragial - R Ataide)

Nesta entrada, como nos é indicada que a direção pela qual o camião percorre a Rua do Alecrim é par, então assumo que existe uma aresta direcional da Rua do Ferragial para a Rua do Alecrim e outra aresta que sai da Rua do Alecrim e vai para a Rua Ataíde.

15806: R do Alecrim (Impar (27-¿53)(-¿): R Ataide - R Ferragial)

Neste caso, a direção é ímpar e, por isso, a direção das arestas é contrária ao caso anterior, isto é, existe uma aresta que vai da rua Ferragial para a rua do Alecrim e outra aresta que vai da Rua do Alecrim para a rua Ataíde.

15808: R Corpo Santo (Ambos (-¿)(10-¿26): Lg Corpo Santo - Tv Corpo Santo)

Como a direção neste exemplo é "Ambos"assumi que existem arestas nas duas

direções, ou seja, o camião pode ir da Lg Corpo Santo para a Rua Corpo Santo, pode ir da Rua Corpo Santo para a TV Corpo Santo, mas também pode ir na direção contrária: Tv Corpo Santo para Rua Corpo Santo e da Rua Corpo Santo para a Lg Corpo Santo.

15832: Tv Santa Catarina, 4-6

Como neste caso não nos é fornecida informação das ruas adjacentes, nem das direções, é criada uma aresta que sai da rua atual para a rua da entrada anterior e outra aresta que sai da rua atual para a rua a seguir.

O parser desenvolvido faz o reconhecimento de todas estas entradas e organiza toda a informação em estruturas de dados tais como dicionários e listas do Python. Como o "mapa"da distribuição dos pontos de recolha pode ser visto como um grafo onde os nodos são os pontos de recolha, foi necessário converter as adjacências entre ruas para arestas entre pontos de recolha. Assim, após organizar toda a informação, o programa verifica, para cada ponto de Recolha, qual a rua onde está localizado. De seguida, procura na lista de adjacências todas as ruas que são acessíveis a partir da mesma. Depois, percorre todos os pontos de recolha e, aqueles que estiverem numa rua acessível a partir da rua do ponto de recolha original, são considerados pontos de recolha adjacentes e é criada uma nova aresta. Estas arestas seguem o seguinte formato:

aresta(15805 , 15808 , 0.1175709600401816).

onde o primeiro valor é o identificador do ponto de recolha de origem e o segundo valor é o identificador do ponto de recolha de destino. O último valor é a distãncia entre as coordenadas dos dois, em km, calculada através do móulo geopy do Python.

Os nodos do grafo são guardados com o seguinte formato:

pontoRecolha(15842 , -9.146144218 , 38.70778398 , 'Misericordia' , 'Tv Carvalho' , [('Lixos', 'CV0090', 90)]).

onde o primeiro valor é o identificador do ponto de recolha, os dois valores seguintes são as coordenadas latitude e longitude, respetivamente, o quarto valor é a Freguesia onde se encontra, o quinto valor é o nome da Rua onde está localizado e o último valor é a lista dos contentores que se encontram nesse ponto de Recolha. Cada "entrada"nesta lista é um tuplo de 3 elementos: tipo de lixo, tipo de contentor e capacidade, respetivamente.

As arestas e os pontos de recolha são guardados em ficheiros módulo do prolog, prontos a serem importados pelo programa principal, cujo formato pode ser visualizado

nos apêndices.

Estratégias de Procura

3.1 Procura não informada

3.1.1 Procura Depth-First

A ideia que está inerente ao método Primeiro em Profundidade é a de tentar avançar de estado para estado até que se encontre a solução. É um método adequado se as opções tomadas forem na direção correta, mas pode ser inadequado se a direção tomada for desadequada. O método Primeiro em Profundidade apresenta a vantagem de ter poucos requisitos em termos de memória. O método é também adequado para problemas que tenham várias soluções pois nesses casos aumenta a possibilidade de se optar por um caminho adequado. O problema do Primeiro em Profundidade é avançar por um caminho que até pode estar a afastar-se da solução e isso é particularmente comum em grafos de grande dimensão como se vai verificar posteriormente. A implementação deste método de procura é a seguinte:

```
dfs(Orig,Dest,Cam,Custo):-dfs(Orig,Dest,[Orig],Cam,Custo).

dfs(Dest,Dest,LA,Cam,0):-reverse(LA,Cam).

dfs(Act,Dest,LA,Cam,Custo):-aresta(Act,X,C1),
                    \+ member(X,LA),
                    dfs(X,Dest,[X|LA],Cam,C2),Custo is C1+C2.
```

3.1.2 Procura Breadth-First

Segundo este método só irá ser possível efetuar a pesquisa no nível N da árvore se todos os nós do nível N-1 tiverem sido explorados. Pode-se dizer que a árvore é explorada transversalmente, derivando daí o nome do método. Garante a obtenção da solução ao nível mais próximo da raiz, o que não quer dizer que seja a melhor solução. Pode requerer muita memória, espaço e tempo para problemas mais complexos. A implementação deste método de procura é a seguinte:

```
bfs(Orig,Dest,Cam):-bfs2(Dest,[[Orig]],Cam).

bfs2(Dest,[[Dest|T]|_],Cam):-reverse([Dest|T],Cam).

bfs2(Dest,[LA|Outros],Cam):-LA=[Act|_],
                            findall([X|LA],(Dest\==Act,aresta(Act,X,_),
                            \+member(X,LA)),Novos),
                            append(Outros,Novos,Todos),
                            bfs2(Dest,Todos,Cam).
```

3.2 Procura informada

3.2.1 Procura Best first

Vamos descrever agora o primeiro método dito "informado" de pesquisa. O método "Primeiro o Melhor" assemelha-se ao Primeiro em Profundidade, a única diferença reside no facto da decisão sobre qual ramo seguir ser feita com base num critério de decisão local. A ideia que está por trás é a de em caso de indecisão sobre qual caminho vamos seguir deve-se optar por aquele que pareça mais promissor. Trata-se de uma decisão feita localmente com base numa informação particular. É então necessário dispor de um valor numérico que avalie o custo ou o ganho que se obtém pelo facto de seguir por um dado caminho. Neste caso, o valor utilizado para comparação é o custo do caminho entre dois pontos que, como foi referido anteriormente, e a soma das distâncias entre os pontos do caminho, calculadas previamente utilizando as coordenadas. A implementação deste método de procura é a seguinte:

```
bestfs(Orig,Dest,Cam,Custo):-bestfs2(Dest,(0,[Orig]),Cam,Custo).

bestfs2(Dest,(Custo,[Dest|T]),Cam,Custo):-!,reverse([Dest|T],Cam).

bestfs2(Dest,(Ca,LA),Cam,Custo):-LA=[Act|_],findall((EstX,CaX,[X|LA]),
                  (aresta(Act,X,CX),\+member(X,LA),estimativa(X,Dest,EstX),
                   CaX is Ca+CX),Novos),
                   sort(Novos,NovosOrd),
                   NovosOrd = [(_,CM,Melhor)|_],
                   bestfs2(Dest,(CM,Melhor),Cam,Custo).
```

3.2.2 Branch and Bound

O Branch and Bound é um método de pesquisa que corresponde ao método "Primeiro o Melhor" com avaliação de transições locais mas com possibilidade de alterar a qualquer momento o nó sobre qual se vai considerar a próxima expansão. Quer isso dizer que o próximo nó que se expande não é obrigatoriamente um descendente do último nó expandido. A comparação entre os nós candidatos à expansão é feita com base no valor acumulado do custo ou do ganho desde a raíz até esses nós. O principal inconveniente do Branch and Bound reside no facto de não ser sensível à distância que um dado nó se encontra da solução. A implementação deste método de procura é a seguinte:

```
bnb(Orig,Dest,Cam,Custo):-bnb2(Dest,[(0,[Orig])],Cam,Custo).

bnb2(Dest,[(Custo,[Dest|T])|_],Cam,Custo):-reverse([Dest|T],Cam).

bnb2(Dest,[(Ca,LA)|Outros],Cam,Custo):-LA=[Act|_],
                        findall((CaX,[X|LA]),
                        (Dest\==Act,aresta(Act,X,CustoX),\+ member(X,LA),
                         CaX is CustoX + Ca),Novos),
                        append(Outros,Novos,Todos),
                        sort(Todos,TodosOrd),
                        bnb2(Dest,TodosOrd,Cam,Custo).
```

3.2.3 Procura A*

Interessa, portanto, juntar o que há de bom no Primeiro o Melhor, ou seja o uso de funções que estimam a distância à solução, com o que há de melhor no Branch and Bound, ou seja o uso de custos acumulados conhecidos e a possibilidade de comutar de um ponto para outro na árvore de pesquisa sem que o novo ponto seja um descendente do primeiro. No fundo isso corresponde ao método A*. No A* consideramos uma função heurística f' tal que:

$$f' = g + h' \tag{3.1}$$

Onde g é o custo conhecido para ir do estado inicial até ao estado do nó que se está a considerar, e que no momento é uma folha ainda não expandida. h' é uma estimativa do custo para ir desse nó até a solução, estando portanto sujeita a erro. Assim, se

estivermos a operar com custos convém que h' seja um minorante do custo real, ao passo que se estivermos a operar com lucros interessa que h' seja um majorante. Neste trabalho, as heurísticas utilizadas são sempre o custo do caminho calculado através da soma das distâncias entre os pontos que o formam. A implementação deste método de procura é a seguinte:

```
aStar(Orig,Dest,Cam,Custo):-aStar2(Dest,[(_,0,[Orig])],Cam,Custo).

aStar2(Dest,[(_,Custo,[Dest|T])|_],Cam,Custo):-reverse([Dest|T],Cam).

aStar2(Dest,[(_,Ca,LA)|Outros],Cam,Custo):-LA=[Act|_],findall((CEX,CaX,[X|LA]),
       (Dest\==Act,aresta(Act,X,CustoX),\+ member(X,LA),
        CaX is CustoX + Ca, estimativa(X,Dest,EstX),
        CEX is CaX +EstX),Novos),
        append(Outros,Novos,Todos),
        sort(Todos,TodosOrd),
        aStar2(Dest,TodosOrd,Cam,Custo).


estimativa(Nodo1,Nodo2,Estimativa):-
pontoRecolha(Nodo1,X1,Y1,_,_,_),
pontoRecolha(Nodo2,X2,Y2,_,_,_),
Estimativa is sqrt((X1-X2)^2+(Y1-Y2)^2).
```

3.2.4 Procura Gulosa

Semelhante à busca em profundidade com busca em profundidade com backtracking. O seu objectivo é:

- Minimizar o custo estimado para atingir um nó.
- Expandir o nó cujo estado é previst como o mais perto do nó final, com base na estimativa feita pela função heurística h.
- Tem de existir um conhecimento prévio dessas funções.
- h (n) = Custo estimado do caminho mais barato desde o estado correspondente ao nó n até a um estado objectivo (final) (soma das distâncias).

A implementação deste método de procura é a seguinte:

```
gulosa(Partida, Destino,Caminho,Custo) :-
    %statistics(runtime,[Start|_]),
    estimativa(Partida,Destino ,Estimativa),
    gulosa2([[Partida]/0/Estimativa],Destino,InvCaminho/Custo/_),
    reverse(InvCaminho, Caminho).
    %,
    %statistics(runtime,[Stop|_]),
    %Runtime is Stop-Start,
    %write("Tempo: "),write(Runtime).

gulosa2(Caminhos, Destino ,Caminho) :-
    melhorCaminhoGul(Caminhos, Caminho),
    Caminho = [Nodo|_]/_/_,
    Nodo == Destino.

gulosa2(Caminhos, Destino ,SolucaoCaminho) :-
    melhorCaminhoGul(Caminhos,MelhorCaminho),
    seleciona(MelhorCaminho, Caminhos, OutrosCaminhos),
    findall(NovoCaminho, adjacenteGul(MelhorCaminho,NovoCaminho,Destino),
    ExpCaminhos),
    append(OutrosCaminhos, ExpCaminhos, NovoCaminhos),
    gulosa2(NovoCaminhos,Destino ,SolucaoCaminho).

melhorCaminhoGul([Caminho], Caminho) :- !.

melhorCaminhoGul([Caminho1/Custo1/Est1,_/_/Est2|Caminhos], MelhorCaminho) :-
    Est1 =< Est2, !,
    melhorCaminhoGul([Caminho1/Custo1/Est1|Caminhos], MelhorCaminho).

melhorCaminhoGul([_|Caminhos], MelhorCaminho) :-
    melhorCaminhoGul(Caminhos, MelhorCaminho).

adjacenteGul([Nodo|Caminho]/Custo/_,
    [ProxNodo,Nodo|Caminho]/NovoCusto/Est,Destino) :-
```

```
    adjacentes(Nodo, ProxNodo, PassoCusto),\+ member(ProxNodo, Caminho),
    NovoCusto is Custo + PassoCusto,
    estimativa(ProxNodo,Destino ,Est).

seleciona(E, [E|Xs], Xs).
seleciona(E, [X|Xs], [X|Ys]) :- seleciona(E, Xs, Ys).
```

Todos estes tipos de pesquisa foram aplicados às funcionalidades descritas a seguir.

Funcionalidades do Programa

4.1 Comparar circuitos

De modo ao sistema ser capaz de recomendar circuitos, é necessário que ele seja capaz de os comparar, para isso foram definidos os seguintes predicados que permitem comparar diferentes circuitos tendo em conta diferentes critérios de avaliação:

4.1.1 Caminho mais rápido

Os seguintes predicados comparam o custo de dois caminho, tendo em conta a distância necessária para percorrer as arestas entre todos os pontos.

```
getCustoCaminho([P1,P2],A):-getCusto(P1,P2,A).
getCustoCaminho([P1,P2|C],Res):-getCusto(P1,P2,A),
                getCustoCaminho([P2|C],Custo2),Res is A+Custo2.
getCustoCaminho(_,_).

caminhoMaisRapido(Caminho1,Caminho2,R,Custo):-getCustoCaminho(Caminho1,Custo1),
                                              getCustoCaminho(Caminho2,Custo2),
caminhoMaisRapido2(Caminho1,Caminho2,Custo1,Custo2,Custo,R).

caminhoMaisRapido2(_,C2,Custo1,Custo2,Custo2,C2):-Custo1>Custo2.
caminhoMaisRapido2(C1,_,Custo1,_,Custo1,C1).
```

4.1.2 Caminho com mais resíduo

Os seguintes predicados comparam a quantidade de resíduos de um determinado tipo que são recolhidos, devolvendo o caminho com onde existe mais resíduo e a respetiva quantidade.

```
getQuantidadeLixo(Ponto,Tipo,Res):-getContentores(Ponto,Contentores),
                findall(X,membro((Tipo,_,X),Contentores),Quantidades),
                somaQuantidades(Quantidades,Res).


getQuantidadeLixoCaminho([],_,0).
getQuantidadeLixoCaminho([H|T],Tipo,Res):-getQuantidadeLixo(H,Tipo,X),
                        getQuantidadeLixoCaminho(T,Tipo,X2),Res is X+X2.


caminhoMaisLixo(C1,C2,Tipo,Res,Qt):-getQuantidadeLixoCaminho(C1,Tipo,L1),
                                getQuantidadeLixoCaminho(C2,Tipo,L2),
                                caminhoMaisLixo2(C1,L1,C2,L2,Res,Qt).

caminhoMaisLixo2(C1,L1,_,L2,C1,L1):-L1>L2.
caminhoMaisLixo2(_,_,C2,L2,C2,L2).
```

4.2 Gerar circuitos

Todas as funcionalidades foram implementadas com todos os métodos de pesquisa explicados anteriormente. Para efeitos de redação, irei referir-me ao método de pesquisa utilizado como método x. As diferentes implementações podem ser verificadas nos apêndices.

4.2.1 Gerar circuitos de recolha que cubram determinados pontos

Uma das funcionalidades é gerar caminhos que cubram uma determinada lista de pontos de recolha. O método utilizado para resolver este problema foi bastante simples. Inicialmente, utiliza-se o método de pesquisa x para procurar o melhor caminho entre

a garagem e o primeiro ponto da lista. De seguida aplica-se o mesmo método para encontrar os caminhos entre todos os pontos adjacentes. Quando chegar ao fim da lista, aplica-se esse método para encontrar o melhor caminho entre o último ponto e o ponto de depósito de resíduos. Finalmente, aplica-se o método x para encontrar o melhor caminho entre o ponto de depósito e a garagem. Juntando todos caminhos gerados numa lista ficámos com o percurso completo tal como foi definido na descrição do problema.

4.2.2 Escolher o circuito mais rápido

De modo a poder gerar o circuito mais rápido, foi necessário implementar um predicado que devolvesse o ponto mais próximo do ponto atual. Para isso foram definidos os seguintes predicados:

```
menorPonto(P1,C1,_,C2,R,C):-C1<C2,R is P1, C is C1.
menorPonto(_,_,P2,C2,R,C):-R is P2, C is C2.

pontoMaisProximo(Visitados,Atual,R,C):-findall((X,Y),aresta(Atual,X,Y),Z),
                                       pontoMaisProximo2(Visitados,Z,R,C,1).


pontoMaisProximo2(Visitados,[(X,Y)|T],R,C,1):-nao(membro(X,Visitados)),
       pontoMaisProximo2(Visitados,T,R2,C2,0),menorPonto(X,Y,R2,C2,R,C).

pontoMaisProximo2(Visitados,[_|T],R,C,1):-pontoMaisProximo2(Visitados,T,R,C,1).

pontoMaisProximo2(Visitados,[(X,Y)|T],R,C,0):-nao(membro(X,Visitados)),
    pontoMaisProximo2(Visitados,T,R2,C2,0),menorPonto(X,Y,R2,C2,R,C).

pontoMaisProximo2(Visitados,[_|T],R,C,0):-pontoMaisProximo2(Visitados,T,R,C,0).

pontoMaisProximo2(_,[],9999,9999,_).
```

Com estes predicados implementados, para gerar o caminho mais rápido, basta procurar sempre o ponto mais próximo do ponto atual (tendo como ponto inicial o ponto onde a garagem se situa). Como a capacidade do camião não está a ser considerada, o camião dirige-se ao ponto de depósito de resíduos quando já percorreu um número

máximo de pontos de recolha a serem visitados, número esse que está definido com um predicado. As diferentes implementações com os vários métodos de pesquisa podem ser verificadas nos apêndices.

4.2.3 Identificar quais os circuitos com mais pontos de recolha

Para saber qual o circuito com mais pontos de recolha foi necessário implementar um predicado que devolvesse o ponto de recolha com menor quantidade de resíduo de um determinado tipo. Para isso foram implementados os seguintes predicados:

```
menorLixo(P1,L1,_,L2,R,L):-L1<L2,R is P1,L is L1.
menorLixo(_,_,P2,L2,R,L):-R is P2, L is L2.

pontoMenosLixo(Visitados,Tipo,R,L):-findall(X,pontoRecolha(X,_,_,_,_,_),Z),
                                    pontoMenosLixo2(Visitados,Tipo,Z,R,L).

pontoMenosLixo2(Visitados,Tipo,[H|T],R,L):-not(membro(H,Visitados)),
                                     getQuantidadeLixo(H,Tipo,L1),
                                     pontoMenosLixo2(Visitados,Tipo,T,R2,L2),
                                     menorLixo(H,L1,R2,L2,R,L).

pontoMenosLixo2(Visitados,Tipo,[_|T],R,L):-pontoMenosLixo2(Visitados,Tipo,T,R,L).

pontoMenosLixo2(_,_,[],9999,9999999).
```

Com estes predicados implementados, para gerar o circuito com mais pontos de recolha, basta simplesmente, escolher sempre o ponto de recolha com menor quantidade de resíduo e aplicar o método de pesquisa de caminho entre dois pontos ao ponto atual e ao ponto encontrado. As diferentes implementações com os vários métodos de pesquisa podem também ser verificados nos apêndices.

Resultados

De modo a poder-se comparar os diferentes métodos de pesquisa, foi necessário diminuir consideravelmente o número de arestas da base de conhecimento (tempora-

riamente). Foi utilizada a funcionalidade de gerar circuito para um dado conjunto de pontos para os seguintes resultados:

Estratégia	Tempo (s)	Custo	Melhor Solução?
DFS	0.5	29.19	Não
BFS	0.5 (s)	0.49	Sim
BestFS	0.1	0.59	Não
BNB	2	0.49	Sim
A*	2	0.49	Sim
Gulosa	0.5	0.58	Não

Conclusão e trabalho futuro

Em conclusão, foram implementadas com sucesso as várias funcionalidades pretendidas, utilizando para tal vários métodos de pesquisa diferentes. Este projeto permitiu confirmar e visualizar as vantagens e desvantagens de cada um destes métodos quer em termos de recursos computacionais como em termos de resultados (circuitos gerados). As maiores dificuldades deste projeto foram: a interpretação e processamento dos dados fornecidos para poderem ser utilizados na geração de um grafo onde se podem aplicar os vários algoritmos desenvolvidos e a grande escala do grafo gerado que tornou complexa a verificação de resultados. Gostava de ter tido tempo de ter em consideração a capacidade máxima do camião para a geração de circuitos, uma vez que essa abordagem aproxima-se muito mais à realidade, mas em contraste, a grande variedade de algoritmos de pesquisa implementados permitiu ter uma maior variedade de resultados.

Apêndices

A.1 Programa principal

```
:-use_module(arestas).
:-use_module(pontosRecolha).
```

```
:- set_prolog_flag(stack_limit, 4_000_000_000).


%definicaoo do ponto em que a garagem se situa
garagem(15805).

%definicao do ponto onde esta o local de deposicao
deposito(15886).

numero_pontos_a_visitar(1).

adjacentes(PontoA,PontoB):-aresta(PontoA,PontoB,_).

adjacentes(PontoA,PontoB,Distancia):-aresta(PontoA,PontoB,
    Distancia).

getCusto(PontoA,PontoB,Custo):-adjacentes(PontoA,PontoB,Custo)
    .

somaQuantidades([],0).
somaQuantidades([H|T],Res):-somaQuantidades(T,R2),Res is H +
    R2.

getContentores(Ponto,R):-pontoRecolha(Ponto,_,_,_,_,R).

getQuantidadeLixo(Ponto,Tipo,Res):-getContentores(Ponto,
    Contentores),findall(X,membro((Tipo,_,X),Contentores),
    Quantidades),somaQuantidades(Quantidades,Res).


getQuantidadeLixoCaminho([],_,0).
getQuantidadeLixoCaminho([H|T],Tipo,Res):-getQuantidadeLixo(H,
    Tipo,X),getQuantidadeLixoCaminho(T,Tipo,X2),Res is X+X2.

getCustoCaminho([P1,P2],A):-getCusto(P1,P2,A).
```

```
getCustoCaminho([P1,P2|C],Res):-getCusto(P1,P2,A),
    getCustoCaminho([P2|C],Custo2),Res is A+Custo2.
getCustoCaminho(_,_).

caminhoMaisRapido(Caminho1,Caminho2,R,Custo):-getCustoCaminho(
    Caminho1,Custo1),getCustoCaminho(Caminho2,Custo2),
    caminhoMaisRapido2(Caminho1,Caminho2,Custo1,Custo2,Custo,R)
    .

caminhoMaisRapido2(_,C2,Custo1,Custo2,Custo2,C2):-Custo1>
    Custo2.
caminhoMaisRapido2(C1,_,Custo1,_,Custo1,C1).



caminhoMaisLixo(C1,C2,Tipo,Res,Qt):-getQuantidadeLixoCaminho(
    C1,Tipo,L1),getQuantidadeLixoCaminho(C2,Tipo,L2),
    caminhoMaisLixo2(C1,L1,C2,L2,Res,Qt).

caminhoMaisLixo2(C1,L1,_,L2,C1,L1):-L1>L2.
caminhoMaisLixo2(_,_,C2,L2,C2,L2).

pontosIguais(A,A).


menorPonto(P1,C1,_,C2,R,C):-C1<C2,R is P1, C is C1.
menorPonto(_,_,P2,C2,R,C):-R is P2, C is C2.

pontoMaisProximo(Visitados,Atual,R,C):-findall((X,Y),aresta(
    Atual,X,Y),Z),pontoMaisProximo2(Visitados,Z,R,C,1).


pontoMaisProximo2(Visitados,[(X,Y)|T],R,C,1):-nao(membro(X,
    Visitados)),pontoMaisProximo2(Visitados,T,R2,C2,0),
    menorPonto(X,Y,R2,C2,R,C).
```

```
pontoMaisProximo2(Visitados,[_|T],R,C,1):-pontoMaisProximo2(
    Visitados,T,R,C,1).

pontoMaisProximo2(Visitados,[(X,Y)|T],R,C,0):-nao(membro(X,
    Visitados)),pontoMaisProximo2(Visitados,T,R2,C2,0),
    menorPonto(X,Y,R2,C2,R,C).

pontoMaisProximo2(Visitados,[_|T],R,C,0):-pontoMaisProximo2(
    Visitados,T,R,C,0).

pontoMaisProximo2(_,[],9999,9999,_).


menorLixo(P1,L1,_,L2,R,L):-L1<L2,R is P1,L is L1.
menorLixo(_,_,P2,L2,R,L):-R is P2, L is L2.

pontoMenosLixo(Visitados,Tipo,R,L):-findall(X,pontoRecolha(X,_
    ,_,_,_,_,_),Z),pontoMenosLixo2(Visitados,Tipo,Z,R,L).

pontoMenosLixo2(Visitados,Tipo,[H|T],R,L):-not(membro(H,
    Visitados)),getQuantidadeLixo(H,Tipo,L1),pontoMenosLixo2(
    Visitados,Tipo,T,R2,L2),menorLixo(H,L1,R2,L2,R,L).

pontoMenosLixo2(Visitados,Tipo,[_|T],R,L):-pontoMenosLixo2(
    Visitados,Tipo,T,R,L).

pontoMenosLixo2(_,_,[],9999,9999999).



dfs(Orig,Dest,Cam,Custo):-dfs(Orig,Dest,[Orig],Cam,Custo).

dfs(Dest,Dest,LA,Cam,0):-reverse(LA,Cam).

dfs(Act,Dest,LA,Cam,Custo):-aresta(Act,X,C1),
                            \+ member(X,LA),
```

```
                                    dfs(X,Dest,[X|LA],Cam,C2),Custo is C1+C2.

bfsS(Orig,Dest,Cam):-bfsS2(Dest,[[Orig]],Cam).

bfsS2(Dest,[[Dest|T]|_],Cam):-reverse([Dest|T],Cam).

bfsS2(Dest,[LA|Outros],Cam):-LA=[Act|_],
                             findall([X|LA],(Dest\==Act,aresta(Act,
                                 X,_),
                             \+member(X,LA)),Novos),
                             append(Outros,Novos,Todos),
                             bfsS2(Dest,Todos,Cam).



bfs(Orig,Dest,Cam,Custo):-bfs2(Dest,[[Orig]],Cam,Custo).

bfs2(Dest,[[Dest|T]|_],Cam,Custo):-reverse([Dest|T],Cam),
    getCustoCaminho(Cam,Custo).

bfs2(Dest,[LA|Outros],Cam,Custo):-LA=[Act|_],
                             findall([X|LA],(Dest\==Act,aresta(Act,
                                 X,_),
                             \+member(X,LA)),Novos),
                             append(Outros,Novos,Todos),
                             bfs2(Dest,Todos,Cam,Custo).



bestfs(Orig,Dest,Cam,Custo):-bestfs2(Dest,(0,[Orig]),Cam,Custo
    ).

bestfs2(Dest,(Custo,[Dest|T]),Cam,Custo):-!,reverse([Dest|T],
    Cam).

bestfs2(Dest,(Ca,LA),Cam,Custo):-LA=[Act|_],findall((EstX,CaX
    ,[X|LA]),
```

```
                (aresta(Act,X,CX),\+member(X,LA),estimativa(X
                    ,Dest,EstX),
                 CaX is Ca+CX),Novos),
                 sort(Novos,NovosOrd),
                 NovosOrd = [(_,CM,Melhor)|_],
                 bestfs2(Dest,(CM,Melhor),Cam,Custo).


bnb(Orig,Dest,Cam,Custo):-bnb2(Dest,[(0,[Orig])],Cam,Custo).

bnb2(Dest,[(Custo,[Dest|T])|_],Cam,Custo):-reverse([Dest|T],
    Cam).

bnb2(Dest,[(Ca,LA)|Outros],Cam,Custo):-LA=[Act|_],
                        findall((CaX,[X|LA]),
                        (Dest\==Act,aresta(Act,X,CustoX),\+
                            member(X,LA),
                         CaX is CustoX + Ca),Novos),
                        append(Outros,Novos,Todos),
                        sort(Todos,TodosOrd),
                        bnb2(Dest,TodosOrd,Cam,Custo).


aStar(Orig,Dest,Cam,Custo):-aStar2(Dest,[(_,0,[Orig])],Cam,
    Custo).

aStar2(Dest,[(_,Custo,[Dest|T])|_],Cam,Custo):-reverse([Dest|T
    ],Cam).

aStar2(Dest,[(_,Ca,LA)|Outros],Cam,Custo):-LA=[Act|_],findall
    ((CEX,CaX,[X|LA]),
          (Dest\==Act,aresta(Act,X,CustoX),\+ member(X,LA),
           CaX is CustoX + Ca, estimativa(X,Dest,EstX),
           CEX is CaX +EstX),Novos),
           append(Outros,Novos,Todos),
           sort(Todos,TodosOrd),
```

```
        aStar2(Dest,TodosOrd,Cam,Custo).


estimativa(Nodo1,Nodo2,Estimativa):-
pontoRecolha(Nodo1,X1,Y1,_,_,_),
pontoRecolha(Nodo2,X2,Y2,_,_,_),
Estimativa is sqrt((X1-X2)^2+(Y1-Y2)^2).


gulosa(Partida, Destino,Caminho,Custo) :-
    %statistics(runtime,[Start|_]),
    estimativa(Partida,Destino ,Estimativa),
    gulosa2([[Partida]/0/Estimativa],Destino,InvCaminho/Custo/
        _),
    reverse(InvCaminho, Caminho).
    %,
    %statistics(runtime,[Stop|_]),
    %Runtime is Stop-Start,
    %write("Tempo: "),write(Runtime).

gulosa2(Caminhos, Destino ,Caminho) :-
    melhorCaminhoGul(Caminhos, Caminho),
    Caminho = [Nodo|_]/_/_,
    Nodo == Destino.

gulosa2(Caminhos, Destino ,SolucaoCaminho) :-
    melhorCaminhoGul(Caminhos,MelhorCaminho),
    seleciona(MelhorCaminho, Caminhos, OutrosCaminhos),
    findall(NovoCaminho, adjacenteGul(MelhorCaminho,
        NovoCaminho,Destino), ExpCaminhos),
    append(OutrosCaminhos, ExpCaminhos, NovoCaminhos),
    gulosa2(NovoCaminhos,Destino ,SolucaoCaminho).

melhorCaminhoGul([Caminho], Caminho) :- !.
```

```
melhorCaminhoGul([Caminho1/Custo1/Est1,_/_/Est2|Caminhos],
    MelhorCaminho) :-
     Est1 =< Est2, !,
     melhorCaminhoGul([Caminho1/Custo1/Est1|Caminhos],
         MelhorCaminho).

melhorCaminhoGul([_|Caminhos], MelhorCaminho) :-
     melhorCaminhoGul(Caminhos, MelhorCaminho).

adjacenteGul([Nodo|Caminho]/Custo/_, [ProxNodo,Nodo|Caminho]/
    NovoCusto/Est,Destino) :-
     adjacentes(Nodo, ProxNodo, PassoCusto),\+ member(ProxNodo,
         Caminho),
     NovoCusto is Custo + PassoCusto,
     estimativa(ProxNodo,Destino ,Est).

seleciona(E, [E|Xs], Xs).
seleciona(E, [X|Xs], [X|Ys]) :- seleciona(E, Xs, Ys).



%Gerar os circuitos de recolha tanto indiferenciada como
    seletiva, caso existam, que
%cubram um determinado territoio;

gerarCircuitoDF2([H,P],Solucao,Custo):-dfs(H,P,S1,C1),
                                       deposito(D),
                                       dfs(P,D,S2,C2),
                                       garagem(G),
                                       dfs(D,G,S3,C3),
                                       append(S1,S2,S4),
                                       append(S4,S3,Solucao),
                                       Custo is C1+C2+C3.


gerarCircuitoDF2([H,P|T],Solucao,Custo):-
```

```
                        dfs(H,P,S1,C1),
                        gerarCircuitoDF2([P|T],S2,C2),
                        append(S1,S2,Solucao),
                        Custo is C1+C2.


gerarCircuitoDF([H,P|T],Solucao,Custo):-garagem(G),dfs(G,H,S1,
    C1),
                                        gerarCircuitoDF2([H,P|
                                            T],S2,C2),
                                        append(S1,S2,Solucao),
                                        Custo is C1+C2.


gerarCircuitoBF2([H,P],Solucao,Custo):-bfs(H,P,S1,C1),
                                       deposito(D),
                                       bfs(P,D,S2,C2),
                                       garagem(G),
                                       bfs(D,G,S3,C3),
                                       append(S1,S2,S4),
                                       append(S4,S3,Solucao),
                                       Custo is C1+C2+C3.


gerarCircuitoBF2([H,P|T],Solucao,Custo):-
                        bfs(H,P,S1,C1),
                        gerarCircuitoBF2([P|T],S2,C2),
                        append(S1,S2,Solucao),
                        Custo is C1+C2.


gerarCircuitoBF([H,P|T],Solucao,Custo):-garagem(G),bfs(G,H,S1,
    C1),
```

```
                                        gerarCircuitoBestFS2([
                                            H,P|T],S2,C2),
                                        append(S1,S2,Solucao),
                                        Custo is C1+C2.

gerarCircuitoBestFS2([H,P],Solucao,Custo):-bestfs(H,P,S1,C1),
                                        deposito(D),
                                        bestfs(P,D,S2,C2),
                                        garagem(G),
                                        bestfs(D,G,S3,C3),
                                        append(S1,S2,S4),
                                        append(S4,S3,Solucao),
                                        Custo is C1+C2+C3.


gerarCircuitoBestFS2([H,P|T],Solucao,Custo):-
                            bestfs(H,P,S1,C1),
                            gerarCircuitoBestFS2([P|T],S2,C2),
                            append(S1,S2,Solucao),
                            Custo is C1+C2.


gerarCircuitoBestFS([H,P|T],Solucao,Custo):-garagem(G),bfs(G,H
    ,S1,C1),
                                        gerarCircuitoBestFS2([
                                            H,P|T],S2,C2),
                                        append(S1,S2,Solucao),
                                        Custo is C1+C2.


gerarCircuitoBNB2([H,P],Solucao,Custo):-bnb(H,P,S1,C1),
                                        deposito(D),
                                        bnb(P,D,S2,C2),
```

```
                                        garagem(G),
                                        bnb(D,G,S3,C3),
                                        append(S1,S2,S4),
                                        append(S4,S3,Solucao),
                                        Custo is C1+C2+C3.


gerarCircuitoBNB2([H,P|T],Solucao,Custo):-
                            bnb(H,P,S1,C1),
                            gerarCircuitoBNB2([P|T],S2,C2),
                            append(S1,S2,Solucao),
                            Custo is C1+C2.


gerarCircuitoBNB([H,P|T],Solucao,Custo):-garagem(G),bnb(G,H,S1
    ,C1),
                                        gerarCircuitoBNB2([H,P
                                            |T],S2,C2),
                                        append(S1,S2,Solucao),
                                        Custo is C1+C2.


gerarCircuitoASTAR2([H,P],Solucao,Custo):-aStar(H,P,S1,C1),
                                        deposito(D),
                                        aStar(P,D,S2,C2),
                                        garagem(G),
                                        aStar(D,G,S3,C3),
                                        append(S1,S2,S4),
                                        append(S4,S3,Solucao),
                                        Custo is C1+C2+C3.
```

```
gerarCircuitoASTAR2([H,P|T],Solucao,Custo):-
                        aStar(H,P,S1,C1),
                        gerarCircuitoASTAR2([P|T],S2,C2),
                        append(S1,S2,Solucao),
                        Custo is C1+C2.


gerarCircuitoASTAR([H,P|T],Solucao,Custo):-garagem(G),aStar(G,
    H,S1,C1),
                                    gerarCircuitoASTAR2([H
                                        ,P|T],S2,C2),
                                    append(S1,S2,Solucao),
                                    Custo is C1+C2.


gerarCircuitoGulosa2([H,P],Solucao,Custo):-gulosa(H,P,S1,C1),
                                    deposito(D),
                                    gulosa(P,D,S2,C2),
                                    garagem(G),
                                    gulosa(D,G,S3,C3),
                                    append(S1,S2,S4),
                                    append(S4,S3,Solucao),
                                    Custo is C1+C2+C3.


gerarCircuitoGulosa2([H,P|T],Solucao,Custo):-
                        gulosa(H,P,S1,C1),
                        gerarCircuitoGulosa2([P|T],S2,C2),
                        append(S1,S2,Solucao),
                        Custo is C1+C2.
```

```
gerarCircuitoGulosa([H,P|T],Solucao,Custo):-garagem(G),gulosa(
    G,H,S1,C1),
                                                gerarCircuitoGulosa2([
                                                    H,P|T],S2,C2),
                                                append(S1,S2,Solucao),
                                                Custo is C1+C2.



gerarTodosCircuitosDF(Lista,Solucoes):-findall((X,Y),
    gerarCircuitoDF(Lista,X,Y),Solucoes).

gerarCircuitoBF([H,P],Solucao):-bfsS(H,P,Solucao).
gerarCircuitoBF([H,P|T],Solucao):-bfsS(H,P,Solucao1),
    gerarCircuitoBF([P|T],Solucao2),append(Solucao2,Solucao1,
    Solucao).

gerarTodosCircuitosBF(Lista,Solucoes):-findall(X,
    gerarCircuitoBF(Lista,X),Solucoes).


gerarCircuitoMaisRapidoDF(Solucao,Custo):-garagem(G),
            pontoMaisProximo([],G,R,_),dfs(G,R,S1,C1),
            gerarCircuitoMaisRapidoDF2(S2,C2,R,[R]),
            append(S1,S2,Solucao),Custo is C1+C2.

gerarCircuitoMaisRapidoDF2(Solucao,Custo,Atual,Visitados):-
                                            tamanho(Visitados,L2),
                                            numero_pontos_a_visitar
                                                (M),
                                            L2>M,
                                            deposito(D),dfs(Atual,D
                                                ,S1,C1),
                                            garagem(G),dfs(D,G,S2,
                                                C2),
```

```
                                        append(S1,S2,Solucao),
                                            Custo is C1+C2.

gerarCircuitoMaisRapidoDF2(Solucao,Custo,Atual,Visitados):-
                                    pontoMaisProximo(Visitados,
                                        Atual,R,C),
                                    dfs(Atual,R,S1,C1),
                                    append([R],Visitados,V),
                                    gerarCircuitoMaisRapidoDF2(
                                        S2,C2,R,V),
                                    append(S1,S2,Solucao),Custo
                                        is C1+C2.


gerarCircuitoMaisPontosDF(Solucao,Custo,Tipo):-garagem(G),
            pontoMenosLixo([],Tipo,R,_),dfs(G,R,S1,C1),
            gerarCircuitoMaisPontosDF2(S2,C2,R,Tipo,[R]),
            append(S1,S2,Solucao),Custo is C1+C2.

gerarCircuitoMaisPontosDF2(Solucao,Custo,Atual,_,Visitados):-
                                        tamanho(Visitados,L2),
                                        numero_pontos_a_visitar
                                            (M),
                                        L2>M,
                                        deposito(D),dfs(Atual,D
                                            ,S1,C1),
                                        garagem(G),dfs(D,G,S2,
                                            C2),
                                        append(S1,S2,Solucao),
                                            Custo is C1+C2.

gerarCircuitoMaisPontosDF2(Solucao,Custo,Atual,Tipo,Visitados)
    :-
                                    pontoMenosLixo(Visitados,
                                        Tipo,R,_),
                                    dfs(Atual,R,S1,C1),
```

```
                                    append([R],Visitados,V),
                                    gerarCircuitoMaisPontosDF2(
                                        S2,C2,R,Tipo,V),
                                    append(S1,S2,Solucao),Custo
                                        is C1+C2.


gerarCircuitoMaisRapidoBF(Solucao,Custo):-garagem(G),
            pontoMaisProximo([],G,R,_),bfs(G,R,S1,C1),
            gerarCircuitoMaisRapidoBF2(S2,C2,R,[R]),
            append(S1,S2,Solucao),Custo is C1+C2.

gerarCircuitoMaisRapidoBF2(Solucao,Custo,Atual,Visitados):-
                                        tamanho(Visitados,L2),
                                        numero_pontos_a_visitar
                                            (M),
                                        L2>M,
                                        deposito(D),bfs(Atual,D
                                            ,S1,C1),
                                        garagem(G),bfs(D,G,S2,
                                            C2),
                                        append(S1,S2,Solucao),
                                            Custo is C1+C2.

gerarCircuitoMaisRapidoBF2(Solucao,Custo,Atual,Visitados):-
                                    pontoMaisProximo(Visitados,
                                        Atual,R,C),
                                    bfs(Atual,R,S1,C1),
                                    append([R],Visitados,V),
                                    gerarCircuitoMaisRapidoBF2(
                                        S2,C2,R,V),
                                    append(S1,S2,Solucao),Custo
                                        is C1+C2.


gerarCircuitoMaisPontosBF(Solucao,Custo,Tipo):-garagem(G),
```

```
            pontoMenosLixo([],Tipo,R,_),bfs(G,R,S1,C1),
            gerarCircuitoMaisPontosBF2(S2,C2,R,Tipo,[R]),
            append(S1,S2,Solucao),Custo is C1+C2.

gerarCircuitoMaisPontosBF2(Solucao,Custo,Atual,_,Visitados):-
                                        tamanho(Visitados,L2),
                                        numero_pontos_a_visitar
                                            (M),
                                        L2>M,
                                        deposito(D),bfs(Atual,D
                                            ,S1,C1),
                                        garagem(G),bfs(D,G,S2,
                                            C2),
                                        append(S1,S2,Solucao),
                                            Custo is C1+C2.

gerarCircuitoMaisPontosBF2(Solucao,Custo,Atual,Tipo,Visitados)
    :-
                                    pontoMenosLixo(Visitados,
                                        Tipo,R,_),
                                    bfs(Atual,R,S1,C1),
                                    append([R],Visitados,V),
                                    gerarCircuitoMaisPontosBF2(
                                        S2,C2,R,Tipo,V),
                                    append(S1,S2,Solucao),Custo
                                        is C1+C2.


gerarCircuitoMaisRapidoBestFS(Solucao,Custo):-garagem(G),
            pontoMaisProximo([],G,R,_),bestfs(G,R,S1,C1),
            gerarCircuitoMaisRapidoBestFS2(S2,C2,R,[R]),
            append(S1,S2,Solucao),Custo is C1+C2.

gerarCircuitoMaisRapidoBestFS2(Solucao,Custo,Atual,Visitados)
    :-
                                        tamanho(Visitados,L2),
```

```
                                        numero_pontos_a_visitar
                                            (M),
                                        L2>M,
                                        deposito(D),bestfs(
                                            Atual,D,S1,C1),
                                        garagem(G),bestfs(D,G,
                                            S2,C2),
                                        append(S1,S2,Solucao),
                                            Custo is C1+C2.

gerarCircuitoMaisRapidoBestFS2(Solucao,Custo,Atual,Visitados)
    :-
                                    pontoMaisProximo(Visitados,
                                        Atual,R,C),
                                    bestfs(Atual,R,S1,C1),
                                    append([R],Visitados,V),
                                    gerarCircuitoMaisRapidoBestFS2
                                        (S2,C2,R,V),
                                    append(S1,S2,Solucao),Custo
                                        is C1+C2.


gerarCircuitoMaisPontosBestFS(Solucao,Custo,Tipo):-garagem(G),
            pontoMenosLixo([],Tipo,R,_),bestfs(G,R,S1,C1),
            gerarCircuitoMaisPontosBestFS2(S2,C2,R,Tipo,[R]),
            append(S1,S2,Solucao),Custo is C1+C2.

gerarCircuitoMaisPontosBestFS2(Solucao,Custo,Atual,_,Visitados
    ):-
                                        tamanho(Visitados,L2),
                                        numero_pontos_a_visitar
                                            (M),
                                        L2>M,
                                        deposito(D),bestfs(
                                            Atual,D,S1,C1),
```

```
                                        garagem(G), bestfs(D,G,
                                            S2,C2),
                                        append(S1,S2,Solucao),
                                            Custo is C1+C2.

gerarCircuitoMaisPontosBestFS2(Solucao,Custo,Atual,Tipo,
    Visitados):-
                                    pontoMenosLixo(Visitados,
                                        Tipo,R,_),
                                    bestfs(Atual,R,S1,C1),
                                    append([R],Visitados,V),
                                    gerarCircuitoMaisPontosBestFS2
                                        (S2,C2,R,Tipo,V),
                                    append(S1,S2,Solucao),Custo
                                        is C1+C2.

gerarCircuitoMaisRapidoBNB(Solucao,Custo):-garagem(G),
            pontoMaisProximo([],G,R,_),bnb(G,R,S1,C1),
            gerarCircuitoMaisRapidoBNB2(S2,C2,R,[R]),
            append(S1,S2,Solucao),Custo is C1+C2.

gerarCircuitoMaisRapidoBNB2(Solucao,Custo,Atual,Visitados):-
                                        tamanho(Visitados,L2),
                                        numero_pontos_a_visitar
                                            (M),
                                        L2>M,
                                        deposito(D),bnb(Atual,D
                                            ,S1,C1),
                                        garagem(G),bnb(D,G,S2,
                                            C2),
                                        append(S1,S2,Solucao),
                                            Custo is C1+C2.

gerarCircuitoMaisRapidoBNB2(Solucao,Custo,Atual,Visitados):-
                                    pontoMaisProximo(Visitados,
                                        Atual,R,C),
```

```
                                        bnb(Atual,R,S1,C1),
                                        append([R],Visitados,V),
                                        gerarCircuitoMaisRapidoBNB2(
                                            S2,C2,R,V),
                                        append(S1,S2,Solucao),Custo
                                            is C1+C2.


gerarCircuitoMaisPontosBNB(Solucao,Custo,Tipo):-garagem(G),
            pontoMenosLixo([],Tipo,R,_),bnb(G,R,S1,C1),
            gerarCircuitoMaisPontosBNB2(S2,C2,R,Tipo,[R]),
            append(S1,S2,Solucao),Custo is C1+C2.

gerarCircuitoMaisPontosBNB2(Solucao,Custo,Atual,_,Visitados):-
                                            tamanho(Visitados,L2),
                                            numero_pontos_a_visitar
                                                (M),
                                            L2>M,
                                            deposito(D),bnb(Atual,D
                                                ,S1,C1),
                                            garagem(G),bnb(D,G,S2,
                                                C2),
                                            append(S1,S2,Solucao),
                                                Custo is C1+C2.

gerarCircuitoMaisPontosBNB2(Solucao,Custo,Atual,Tipo,Visitados
    ):-
                                        pontoMenosLixo(Visitados,
                                            Tipo,R,_),
                                        bnb(Atual,R,S1,C1),
                                        append([R],Visitados,V),
                                        gerarCircuitoMaisPontosBNB2(
                                            S2,C2,R,Tipo,V),
                                        append(S1,S2,Solucao),Custo
                                            is C1+C2.
```

```
gerarCircuitoMaisRapidoAStar(Solucao,Custo):-garagem(G),
            pontoMaisProximo([],G,R,_),aStar(G,R,S1,C1),
            gerarCircuitoMaisRapidoAStar2(S2,C2,R,[R]),
            append(S1,S2,Solucao),Custo is C1+C2.

gerarCircuitoMaisRapidoAStar2(Solucao,Custo,Atual,Visitados):-
                                        tamanho(Visitados,L2),
                                        numero_pontos_a_visitar
                                            (M),
                                        L2>M,
                                        deposito(D),aStar(Atual
                                            ,D,S1,C1),
                                        garagem(G),aStar(D,G,S2
                                            ,C2),
                                        append(S1,S2,Solucao),
                                            Custo is C1+C2.

gerarCircuitoMaisRapidoAStar2(Solucao,Custo,Atual,Visitados):-
                                    pontoMaisProximo(Visitados,
                                        Atual,R,C),
                                    aStar(Atual,R,S1,C1),
                                    append([R],Visitados,V),
                                    gerarCircuitoMaisRapidoAStar2
                                        (S2,C2,R,V),
                                    append(S1,S2,Solucao),Custo
                                        is C1+C2.


gerarCircuitoMaisPontosAStar(Solucao,Custo,Tipo):-garagem(G),
            pontoMenosLixo([],Tipo,R,_),aStar(G,R,S1,C1),
            gerarCircuitoMaisPontosAStar2(S2,C2,R,Tipo,[R]),
            append(S1,S2,Solucao),Custo is C1+C2.

gerarCircuitoMaisPontosAStar2(Solucao,Custo,Atual,_,Visitados)
    :-
```

```
                                        tamanho(Visitados,L2),
                                        numero_pontos_a_visitar
                                            (M),
                                        L2>M,
                                        deposito(D),aStar(Atual
                                            ,D,S1,C1),
                                        garagem(G),aStar(D,G,S2
                                            ,C2),
                                        append(S1,S2,Solucao),
                                            Custo is C1+C2.

gerarCircuitoMaisPontosAStar2(Solucao,Custo,Atual,Tipo,
    Visitados):-
                                    pontoMenosLixo(Visitados,
                                        Tipo,R,_),
                                    aStar(Atual,R,S1,C1),
                                    append([R],Visitados,V),
                                    gerarCircuitoMaisPontosAStar2
                                        (S2,C2,R,Tipo,V),
                                    append(S1,S2,Solucao),Custo
                                        is C1+C2.

gerarCircuitoMaisRapidoGulosa(Solucao,Custo):-garagem(G),
            pontoMaisProximo([],G,R,_),gulosa(G,R,S1,C1),
            gerarCircuitoMaisRapidoGulosa2(S2,C2,R,[R]),
            append(S1,S2,Solucao),Custo is C1+C2.

gerarCircuitoMaisRapidoGulosa2(Solucao,Custo,Atual,Visitados)
    :-
                                        tamanho(Visitados,L2),
                                        numero_pontos_a_visitar
                                            (M),
                                        L2>M,
                                        deposito(D),gulosa(
                                            Atual,D,S1,C1),
```

```
                                        garagem(G),gulosa(D,G,
                                            S2,C2),
                                        append(S1,S2,Solucao),
                                            Custo is C1+C2.

gerarCircuitoMaisRapidoGulosa2(Solucao,Custo,Atual,Visitados)
    :-
                                    pontoMaisProximo(Visitados,
                                        Atual,R,C),
                                    gulosa(Atual,R,S1,C1),
                                    append([R],Visitados,V),
                                    gerarCircuitoMaisRapidoGulosa2
                                        (S2,C2,R,V),
                                    append(S1,S2,Solucao),Custo
                                        is C1+C2.


gerarCircuitoMaisPontosGulosa(Solucao,Custo,Tipo):-garagem(G),
            pontoMenosLixo([],Tipo,R,_),gulosa(G,R,S1,C1),
            gerarCircuitoMaisPontosGulosa2(S2,C2,R,Tipo,[R]),
            append(S1,S2,Solucao),Custo is C1+C2.

gerarCircuitoMaisPontosGulosa2(Solucao,Custo,Atual,_,Visitados
    ):-
                                        tamanho(Visitados,L2),
                                        numero_pontos_a_visitar
                                            (M),
                                        L2>M,
                                        deposito(D),gulosa(
                                            Atual,D,S1,C1),
                                        garagem(G),gulosa(D,G,
                                            S2,C2),
                                        append(S1,S2,Solucao),
                                            Custo is C1+C2.
```

```
gerarCircuitoMaisPontosGulosa2(Solucao,Custo,Atual,Tipo,
    Visitados):-
                                        pontoMenosLixo(Visitados,
                                            Tipo,R,_),
                                        gulosa(Atual,R,S1,C1),
                                        append([R],Visitados,V),
                                        gerarCircuitoMaisPontosGulosa2
                                            (S2,C2,R,Tipo,V),
                                        append(S1,S2,Solucao),Custo
                                            is C1+C2.


%predicados auxiliares
%
tamanho([]       , 0 ).
tamanho([_|Xs]  , L ) :- tamanho(Xs,N)  , L is N+1 .

nao( Questao ) :-
    Questao, !, fail.
nao( Questao ).

membro(X, [X|_]).
membro(X, [_|Xs]):-
        membro(X, Xs).

inverso([],Z,Z).

inverso([H|T],Z,Acc) :- inverso(T,Z,[H|Acc]).
```

A.2 Parser

A.2.1 Lexer

```
import ply.lex as lex
```

```
reserved = {
    'Par ': 'PAR',
    'Impar ': 'IMPAR',
    'Ambos ': 'AMBOS',
}

tokens = ['FLOAT', 'INT', 'WORD', 'LIXO', 'SEQ'] + list(
   reserved.values())

literals = [',', ':', ')', '(', '-']


def t_WORD(t):
    r'\s*[a-zA-Z][\.\d\w\s]*'
    t.type = reserved.get(t.value, 'WORD')    # Check for
       reserved words
    return t


t_FLOAT = r'[\-+]?\d+\.\d+'
t_INT = r'\s*\d+\s*'
t_LIXO = r'\(\d*\->\d*\)\(\d*\->\d*\)'
t_SEQ = r'\s*(\d+\-)+\d+'


def t_error(t):
    #global l
    print("Carater ilegal: ", t.value[0])
    #print('Linha :' + str(l))
    t.lexer.skip(1)


t_ignore = "\t\n"/"
lexer = lex.lex()
```

```
# PARA TESTAR O ANALISADOR LEXICO:
"""
l = 0
import sys
with open('dataset.csv') as input:
    for linha in input:
        lexer.input(linha)
        for tok in lexer:
            print(tok)
        l += 1
"""
```

A.2.2 Gramática

```
import ply.yacc as yacc
from dataset_lex import tokens
from dataset_lex import literals
import sys
import re
from geopy.distance import distance

last_street = ''
info = []
pontoId = 0


def deleteNewLine(string):
    return int(re.sub(r'\n', '', string))


def strip_one_space(s):
    if s.endswith(" "):
        s = s[:-1]
    if s.startswith(" "):
        s = s[1:]
    return s
```

```
def getAdjacencias(rua, arestas):
    res = []
    for aresta in arestas:
        if aresta[0] == rua:
            res.append(aresta[1])
    return res


def exportArestas(arestas, pontos_recolha):
    arestas_pontos = []
    for i in pontos_recolha.items():
        adjacencias = getAdjacencias(i[1][3], arestas)
        for j in pontos_recolha.items():
            if j[1][3] in adjacencias:
                d = distance((i[1][0], i[1][1]), (j[1][0], j
                    [1][1])).km
                if (i[0], j[0], d) not in arestas_pontos:
                    arestas_pontos.append((i[0], j[0], d))

    with open("arestas.pl", 'w') as f:
        f.write(':- module(arestas, [aresta/3]).\n')
        for aresta in arestas_pontos:
            f.write('aresta( ' + str(aresta[0]) + ' , ' +
                    str(aresta[1]) + ' , ' + str(aresta[2]) +
                        ' ).\n')


def exportPontosRecolha(pontos_recolha):
    with open("pontosRecolha.pl", 'w') as f:
        f.write(':- module(pontosRecolha, [pontoRecolha/6]).\n
            ')
        for i in pontos_recolha.items():
            f.write('pontoRecolha( ' + i[0] + ' , ' + i[1][0]
                + \
```

```
            ' , ' + i[1][1] + ' , \'' + i[1][2] + '\' , \'' +
               \
            strip_one_space(i[1][3]) + '\' , ' + str(i[1][4])
               + ').\n')



def contains(list, elem):
    for i in range(len(list)):
        if list[i] == elem:
            return True
    return False


def addIfNew(list, elem):
    if not contains(list, elem) and elem[0] != elem[1]
    and elem[0] != '' and elem[1] != '':
        list.append(elem)


def updateContentor():
    global last_street, info, pontoId
    if pontoId in parser.pontos_recolha:
        aux = parser.pontos_recolha.get(pontoId)
        aux2 = aux[4]
        if not contains(aux2, info[4][0]):
            aux2.append(info[4][0])
            aux[4] = aux2
            parser.pontos_recolha.update({pontoId: aux})
    else:
        parser.pontos_recolha.update({pontoId: info})


def p_Entry(p):
    "Entry : FLOAT ',' FLOAT ',' INT ',' WORD ',' RUAS ','
    WORD ',' WORD ',' INT ',' INT ',' INT"
```

```
    global info
    p[0] = p[1] + p[3] + p[5] + p[7] + p[9] + \
        p[11] + p[13] + p[15] + p[17] + p[19]
    info[0] = p[1]
    info[1] = p[3]
    info[2] = p[7]
    info[4] = [(p[11], p[13], deleteNewLine(p[19]))]

    updateContentor()


def p_RUAS_Par(p):
    "RUAS : INT ':' WORD '(' PAR LIXO ':' WORD '-' WORD ')'"
    global last_street, info, pontoId
    p[0] = 'ruas par'
    addIfNew(parser.arestas, (p[8], p[3]))
    addIfNew(parser.arestas, (p[3], p[10]))
    addIfNew(parser.arestas, (p[3], last_street))
    addIfNew(parser.arestas, (last_street, p[3]))
    last_street = p[3]
    info = [0, 0, 0, p[3], [], 0]
    pontoId = p[1]


def p_RUAS_Impar(p):
    "RUAS : INT ':' WORD '(' IMPAR LIXO ':' WORD '-' WORD ')'"
    global last_street, info, pontoId
    p[0] = 'ruas impar'
    addIfNew(parser.arestas, (p[10], p[3]))
    addIfNew(parser.arestas, (p[3], p[8]))
    addIfNew(parser.arestas, (p[3], last_street))
    addIfNew(parser.arestas, (last_street, p[3]))
    last_street = p[3]
    info = [0, 0, 0, p[3], []]
    pontoId = p[1]
```

```
def p_RUAS_Ambos(p):
    "RUAS : INT ':' WORD '(' AMBOS LIXO ':' WORD '-' WORD ')'"
    global last_street, info, pontoId
    p[0] = 'ruas ambos'
    addIfNew(parser.arestas, (p[8], p[3]))
    addIfNew(parser.arestas, (p[3], p[10]))
    addIfNew(parser.arestas, (p[10], p[3]))
    addIfNew(parser.arestas, (p[3], p[8]))
    addIfNew(parser.arestas, (p[3], last_street))
    addIfNew(parser.arestas, (last_street, p[3]))

    last_street = p[3]
    info = [0, 0, 0, p[3], []]
    pontoId = p[1]


def p_RUAS_rua_rua(p):
    "RUAS : INT ':' WORD ',' WORD"
    global last_street, info, pontoId
    p[0] = 'rua'
    addIfNew(parser.arestas, (p[3], p[5]))
    addIfNew(parser.arestas, (p[3], last_street))
    addIfNew(parser.arestas, (last_street, p[3]))
    last_street = p[3]
    info = [0, 0, 0, p[3], []]
    pontoId = p[1]


def p_RUAS_rua_int(p):
    "RUAS : INT ':' WORD ',' INT"
    global last_street, info, pontoId
    p[0] = 'rua'
    addIfNew(parser.arestas, (p[3], last_street))
    addIfNew(parser.arestas, (last_street, p[3]))
    last_street = p[3]
```

```
    info = [0, 0, 0, p[3], []]
    pontoId = p[1]


def p_RUAS_rua_seq(p):
    "RUAS : INT ':' WORD ',' SEQ"
    global last_street, info, pontoId
    p[0] = 'rua'
    addIfNew(parser.arestas, (p[3], last_street))
    addIfNew(parser.arestas, (last_street, p[3]))
    last_street = p[3]
    info = [0, 0, 0, p[3], []]
    pontoId = p[1]


def p_RUAS_rua_INT_WORD(p):
    "RUAS : INT ':' WORD ',' INT WORD"
    global last_street, info, pontoId
    p[0] = 'rua'
    addIfNew(parser.arestas, (p[3], last_street))
    addIfNew(parser.arestas, (last_street, p[3]))
    addIfNew(parser.arestas, (p[3], p[5]))
    last_street = p[3]
    info = [0, 0, 0, p[3], []]
    pontoId = p[1]


def p_error(p):
    global l
    print("Syntax error in input: ", p)
    print('linha ' + str(l))


parser = yacc.yacc()
```

```
# id -> [latitude,longitude,Freguesia,rua,[(tipo residuo,
    tipo_contentor,litros)...]]
parser.pontos_recolha = {}


# rua -> rua (significa que o camiao viaja no sentido esquerda
    para a direita)
parser.arestas = []
total = 0
l = 0


with open('dataset.csv') as input:
    for linha in input:
        result = parser.parse(linha)
        if result:
            # print('entry valida: \n\n' + result)
            total += 1
        l += 1

print('TOTAL ENTRIES VALIDAS: ' + str(total))
# print(parser.arestas)
# print(str(len(parser.arestas)))
# print(parser.pontos_recolha)
# print(str(len(parser.pontos_recolha)))
exportArestas(parser.arestas, parser.pontos_recolha)
exportPontosRecolha(parser.pontos_recolha)
```

A.3 Base de Conhecimento

A.3.1 Pontos de Recolha

```
:- module(pontosRecolha, [pontoRecolha/6]).
pontoRecolha( 15805 , -9.143308809 , 38.70807879 , '
    Misericordia' , 'R do Alecrim' , [('Lixos', 'CV0090', 90),
```

```
    ('Lixos', 'CV0240', 1680), ('Papel e Cartao', 'CV0240',
    1440), ('Papel e Cartao', 'CV0090', 90)]).
pontoRecolha( 15806 , -9.143377778 , 38.70807819 , '
    Misericordia' , 'R do Alecrim' , [('Lixos', 'CV0240', 240),
     ('Lixos', 'CV0140', 840), ('Lixos', 'CV0120', 120), ('
    Lixos', 'CV0090', 180), ('Papel e Cartao', 'CV0140', 280),
    ('Papel e Cartao', 'CV0090', 90), ('Papel e Cartao', '
    CA0090', 90)]).
pontoRecolha( 15807 , -9.143481807 , 38.70730262 , '
    Misericordia' , 'R do Alecrim' , [('Lixos', 'CV0240', 240),
     ('Lixos', 'CV0140', 420), ('Papel e Cartao', 'CV0240',
    240), ('Papel e Cartao', 'CV0140', 140)]).
pontoRecolha( 15808 , -9.142550987 , 38.70732868 , '
    Misericordia' , 'R Corpo Santo' , [('Lixos', 'CV0140', 280)
    , ('Lixos', 'CV0240', 960)]).
pontoRecolha( 15809 , -9.142765961 , 38.70708361 , '
    Misericordia' , 'Tv Corpo Santo' , [('Lixos', 'CV0140',
    280), ('Lixos', 'CV0120', 120)]).
pontoRecolha( 15810 , -9.142622569 , 38.70669752 , '
    Misericordia' , 'Tv Corpo Santo' , [('Lixos', 'CV0240',
    960), ('Papel e Cartao', 'CV0240', 240)]).
pontoRecolha( 15811 , -9.142408356 , 38.70699663 , '
    Misericordia' , 'R Bernardino da Costa' , [('Lixos', '
    CV0140', 420), ('Lixos', 'CV0240', 960), ('Papel e Cartao',
     'CV0240', 240)]).
pontoRecolha( 15812 , -9.143050155 , 38.70685596 , '
    Misericordia' , 'R Bernardino da Costa' , [('Lixos', '
    CV0140', 700), ('Lixos', 'CV0240', 720), ('Lixos', 'CV0120
    ', 360)]).
pontoRecolha( 15813 , -9.151251124 , 38.70877545 , '
    Misericordia' , 'Lg Conde Barao' , [('Lixos', 'CV0240',
    2400), ('Lixos', 'CV0140', 980), ('Lixos', 'CV0090', 90)]).
pontoRecolha( 15814 , -9.151789464 , 38.70863563 , '
    Misericordia' , 'Lg Conde Barao' , [('Lixos', 'CV0120',
    240), ('Lixos', 'CV0140', 700), ('Lixos', 'CV0090', 90), ('
    Lixos', 'CV0240', 480)]).
```

```
pontoRecolha( 15815 , -9.152018979 , 38.70860661 , '
    Misericordia' , 'Lg Conde Barao' , [('Lixos', 'CV0120',
    120), ('Lixos', 'CV0140', 280), ('Lixos', 'CV0240', 240)]).
pontoRecolha( 15818 , -9.149105527 , 38.70907338 , '
    Misericordia' , 'Tv Marques de Sampaio' , [('Lixos', '
    CV0240', 240)]).
pontoRecolha( 15819 , -9.147649761 , 38.70856359 , '
    Misericordia' , 'R da Boavista' , [('Lixos', 'CV0140',
    1260), ('Lixos', 'CV0120', 240), ('Lixos', 'CV0240', 720),
    ('Lixos', 'CV0090', 180), ('Lixos', 'CV1100', 1100)]).
pontoRecolha( 15820 , -9.14857178 , 38.70872672 , '
    Misericordia' , 'R da Boavista' , [('Lixos', 'CV0140', 840)
    , ('Lixos', 'CV0240', 480), ('Lixos', 'CV0090', 90)]).
pontoRecolha( 15821 , -9.149113446 , 38.7088211 , '
    Misericordia' , 'R da Boavista' , [('Lixos', 'CV0140', 280)
    , ('Lixos', 'CV0240', 480), ('Lixos', 'CV0090', 90)]).
pontoRecolha( 15822 , -9.149493546 , 38.70887183 , '
    Misericordia' , 'R da Boavista' , [('Lixos', 'CV0090', 180)
    , ('Lixos', 'CV0120', 120), ('Lixos', 'CV0140', 140), ('
    Lixos', 'CV0240', 720)]).
pontoRecolha( 15823 , -9.150183884 , 38.70891086 , '
    Misericordia' , 'R da Boavista' , [('Lixos', 'CV0140',
    1680), ('Lixos', 'CV0090', 270), ('Lixos', 'CV0240', 3120),
     ('Lixos', 'CV0120', 120)]).
pontoRecolha( 15824 , -9.150793117 , 38.70890555 , '
    Misericordia' , 'R da Boavista' , [('Lixos', 'CV0140', 280)
    , ('Lixos', 'CV0120', 120), ('Lixos', 'CV0090', 90)]).
pontoRecolha( 15825 , -9.152060348 , 38.70828196 , '
    Misericordia' , 'Tv do Cais do Tojo' , [('Lixos', 'CV0140',
     140)]).
pontoRecolha( 15827 , -9.152472114 , 38.70813424 , '
    Misericordia' , 'R Cais do Tojo' , [('Lixos', 'CV0240',
    1920), ('Lixos', 'CV0120', 120)]).
pontoRecolha( 15828 , -9.151783576 , 38.70822132 , '
    Misericordia' , 'R Cais do Tojo' , [('Lixos', 'CV0090', 90)
    , ('Lixos', 'CV0120', 120), ('Lixos', 'CV0140', 420), ('
```

```
   Lixos', 'CV0240', 480)]).
pontoRecolha( 15830 , -9.151544743 , 38.70840356 , '
   Misericordia' , 'Bqr do Duro' , [('Lixos', 'CV0240', 1200)
   ]).
pontoRecolha( 15831 , -9.151388528 , 38.70792751 , '
   Misericordia' , 'Bqr do Duro' , [('Lixos', 'CV0120', 240),
   ('Lixos', 'CV0090', 90), ('Lixos', 'CV0140', 280), ('Lixos
   ', 'CV0240', 720)]).
pontoRecolha( 15832 , -9.147513949 , 38.71033931 , '
   Misericordia' , 'Tv Santa Catarina' , [('Lixos', 'CV0240',
   240), ('Vidro', 'CV0240', 240)]).
pontoRecolha( 15833 , -9.148566019 , 38.70994283 , '
   Misericordia' , 'R Ferreiros a Santa Catarina' , [('Lixos',
    'CV0240', 720)]).
pontoRecolha( 15834 , -9.150464329 , 38.70680059 , '
   Misericordia' , 'R Instituto Industrial' , [('Lixos', '
   CV0140', 140), ('Lixos', 'CV0240', 1920)]).
pontoRecolha( 15836 , -9.15054864 , 38.70788079 , '
   Misericordia' , 'R Instituto Industrial' , [('Lixos', '
   CV0240', 5520), ('Lixos', 'CV0140', 140)]).
pontoRecolha( 15838 , -9.148518124 , 38.70980813 , '
   Misericordia' , 'R Santa Catarina' , [('Lixos', 'CV0240',
   960)]).
pontoRecolha( 15841 , -9.147044252 , 38.70801938 , '
   Misericordia' , 'R Moeda' , [('Lixos', 'CV0240', 2160), ('
   Lixos', 'CV0140', 700)]).
pontoRecolha( 15842 , -9.146144218 , 38.70778398 , '
   Misericordia' , 'Tv Carvalho' , [('Lixos', 'CV0090', 90)]).
pontoRecolha( 15843 , -9.145639847 , 38.70788744 , '
   Misericordia' , 'Tv Carvalho' , [('Lixos', 'CV0090', 90),
   ('Lixos', 'CV0140', 560), ('Lixos', 'CV0240', 480), ('Papel
    e Cartao', 'CV0090', 90), ('Papel e Cartao', 'CV0140',
   280)]).
pontoRecolha( 15844 , -9.145988636 , 38.70816366 , '
   Misericordia' , 'Tv Carvalho' , [('Lixos', 'CV0140', 140)])
   .
```

```
pontoRecolha( 15845 , -9.151993526 , 38.70762498 , '
    Misericordia' , 'R Dom Luis I' , [('Lixos', 'CV0120', 120),
     ('Lixos', 'CV0240', 4800)]).
pontoRecolha( 15846 , -9.150935625 , 38.70760718 , '
    Misericordia' , 'R Dom Luis I' , [('Lixos', 'CV0240', 1200)
    ]).
pontoRecolha( 15847 , -9.149935581 , 38.7076159 , '
    Misericordia' , 'R Dom Luis I' , [('Lixos', 'CV0240', 4320)
    , ('Lixos', 'CV0120', 120)]).
pontoRecolha( 15848 , -9.148290555 , 38.70754014 , '
    Misericordia' , 'R Dom Luis I' , [('Lixos', 'CV0240', 3120)
    , ('Lixos', 'CV1100', 4400), ('Lixos', 'CV0120', 120), ('
    Lixos', 'CV0090', 90)]).
pontoRecolha( 15849 , -9.147146202 , 38.70709969 , '
    Misericordia' , 'Pc Dom Luis I' , [('Lixos', 'CV0240', 960)
    ]).
pontoRecolha( 15850 , -9.148353992 , 38.71039507 , '
    Misericordia' , 'Tv Condessa do Rio' , [('Lixos', 'CV0240',
     720), ('Embalagens', 'CV0240', 240)]).
pontoRecolha( 15851 , -9.146715901 , 38.70756283 , '
    Misericordia' , 'Pc Dom Luis I' , [('Lixos', 'CV0240', 240)
    , ('Lixos', 'CV0140', 280), ('Papel e Cartao', 'CV0140',
    280)]).
pontoRecolha( 15852 , -9.144930015 , 38.70728106 , '
    Misericordia' , 'R Ribeira Nova' , [('Lixos', 'CV0140',
    140), ('Lixos', 'CV0120', 120), ('Lixos', 'CV0090', 90), ('
    Lixos', 'CV0240', 240), ('Papel e Cartao', 'CV0240', 240)])
    .
pontoRecolha( 15853 , -9.145853665 , 38.7075613 , '
    Misericordia' , 'R Ribeira Nova' , [('Lixos', 'CV0120',
    120), ('Lixos', 'CV0140', 420), ('Lixos', 'CV0090', 90), ('
    Lixos', 'CV0240', 480), ('Papel e Cartao', 'CV0340', 680),
    ('Papel e Cartao', 'CV0090', 90), ('Papel e Cartao', '
    CV0240', 720)]).
pontoRecolha( 15855 , -9.14350797 , 38.70752758 , '
    Misericordia' , 'R Sao Paulo' , [('Lixos', 'CV0240', 720),
```

```
    ('Lixos', 'CV0120', 120), ('Lixos', 'CV0140', 700)]).
pontoRecolha( 15856 , -9.144154727 , 38.70773817 , '
    Misericordia' , 'R Sao Paulo' , [('Lixos', 'CV0240', 960),
    ('Lixos', 'CV0140', 280), ('Lixos', 'CV0120', 120)]).
pontoRecolha( 15857 , -9.144188704 , 38.70770185 , '
    Misericordia' , 'R Sao Paulo' , [('Lixos', 'CV0240', 240),
    ('Lixos', 'CV0140', 140), ('Papel e Cartao', 'CV0140', 140)
    ]).
pontoRecolha( 15858 , -9.14351883 , 38.70748245 , '
    Misericordia' , 'R Sao Paulo' , [('Lixos', 'CV0090', 90),
    ('Lixos', 'CV0140', 420), ('Lixos', 'CV0240', 240)]).
pontoRecolha( 15859 , -9.144754745 , 38.70789511 , '
    Misericordia' , 'R Sao Paulo' , [('Lixos', 'CV0090', 270),
    ('Lixos', 'CV0140', 700), ('Lixos', 'CV0240', 480), ('Papel
     e Cartao', 'CV0140', 140)]).
pontoRecolha( 15860 , -9.145528588 , 38.70814963 , '
    Misericordia' , 'R Sao Paulo' , [('Lixos', 'CV0090', 270),
    ('Lixos', 'CV0240', 720), ('Lixos', 'CV0140', 1540), ('
    Lixos', 'CV0120', 240), ('Papel e Cartao', 'CV0240', 480)])
    .
pontoRecolha( 15861 , -9.146060151 , 38.70834319 , '
    Misericordia' , 'R Sao Paulo' , [('Lixos', 'CV0140', 700),
    ('Lixos', 'CV0090', 180), ('Lixos', 'CV0120', 120)]).
pontoRecolha( 15862 , -9.146348797 , 38.70843077 , '
    Misericordia' , 'R Sao Paulo' , [('Lixos', 'CV0090', 90),
    ('Lixos', 'CV0140', 280)]).
pontoRecolha( 15863 , -9.14664843 , 38.70848221 , '
    Misericordia' , 'R Sao Paulo' , [('Lixos', 'CV0240', 240),
    ('Lixos', 'CV0140', 140)]).
pontoRecolha( 15864 , -9.146958664 , 38.70847051 , '
    Misericordia' , 'R Sao Paulo' , [('Lixos', 'CV0240', 960),
    ('Lixos', 'CV0140', 560), ('Lixos', 'CV0090', 90), ('Lixos
    ', 'CV0120', 120)]).
pontoRecolha( 15865 , -9.144288217 , 38.70742174 , '
    Misericordia' , 'R Nova do Carvalho' , [('Lixos', 'CV0120',
     120), ('Lixos', 'CV0140', 280), ('Lixos', 'CV0240', 960),
```

```
    ('Papel e Cartao', 'CV0140', 140), ('Papel e Cartao', '
    CV0240', 240)]).
pontoRecolha( 15866 , -9.143814518 , 38.7072547 , '
    Misericordia' , 'R Nova do Carvalho' , [('Lixos', 'CV0140',
     140), ('Lixos', 'CV0240', 2640), ('Papel e Cartao', '
    CV0240', 480)]).
pontoRecolha( 15867 , -9.143192661 , 38.70717901 , '
    Misericordia' , 'R Nova do Carvalho' , [('Lixos', 'CV0090',
     180), ('Lixos', 'CV0240', 720), ('Lixos', 'CV0140', 420)])
    .
pontoRecolha( 15868 , -9.144027836 , 38.70689253 , '
    Misericordia' , 'R Remolares' , [('Lixos', 'CV0090', 90),
    ('Lixos', 'CV0240', 1200), ('Lixos', 'CV0140', 700), ('
    Papel e Cartao', 'CV0140', 280)]).
pontoRecolha( 15869 , -9.144478925 , 38.7070868 , '
    Misericordia' , 'R Remolares' , [('Lixos', 'CV0140', 560),
    ('Lixos', 'CV0120', 240)]).
pontoRecolha( 15871 , -9.144416495 , 38.70673604 , '
    Misericordia' , 'Tv dos Remolares' , [('Lixos', 'CV0240',
    240)]).
pontoRecolha( 15873 , -9.144169838 , 38.70717956 , '
    Misericordia' , 'Tv dos Remolares' , [('Lixos', 'CV0140',
    280), ('Lixos', 'CV0240', 240), ('Lixos', 'CV0120', 120)]).
pontoRecolha( 15875 , -9.14401336 , 38.70749619 , '
    Misericordia' , 'Tv dos Remolares' , [('Lixos', 'CV0090',
    90), ('Lixos', 'CV0240', 240), ('Lixos', 'CV0140', 140)]).
pontoRecolha( 15876 , -9.144047329 , 38.70582944 , '
    Misericordia' , 'Cais do Sodre' , [('Lixos', 'CV0240',
    1680)]).
pontoRecolha( 15877 , -9.143242333 , 38.70580939 , '
    Misericordia' , 'Cais do Sodre' , [('Lixos', 'CV0240', 240)
    ]).
pontoRecolha( 15878 , -9.142814171 , 38.70642562 , '
    Misericordia' , 'Cais do Sodre' , [('Lixos', 'CV0240', 240)
    , ('Lixos', 'CV0140', 140)]).
pontoRecolha( 15879 , -9.142229849 , 38.70656579 , '
```

```
    Misericordia' , 'Cais do Sodre' , [('Lixos', 'CV0140', 280)
    , ('Papel e Cartao', 'CV0140', 140)]).
pontoRecolha( 15880 , -9.146220062 , 38.70826974 , '
    Misericordia' , 'Bc da Moeda' , [('Lixos', 'CV0090', 90)]).
pontoRecolha( 15881 , -9.149564432 , 38.70900633 , '
    Misericordia' , 'Pto Galega' , [('Lixos', 'CV0240', 240)]).
pontoRecolha( 15882 , -9.150599493 , 38.70903335 , '
    Misericordia' , 'Bc da Boavista' , [('Lixos', 'CV0140',
    140), ('Lixos', 'CV0090', 90)]).
pontoRecolha( 15883 , -9.144620547 , 38.7073468 , '
    Misericordia' , 'Tv Ribeira Nova' , [('Lixos', 'CV0090',
    90)]).
pontoRecolha( 15884 , -9.144764004 , 38.70692219 , '
    Misericordia' , 'Pc Ribeira Nova' , [('Lixos', 'CV0240',
    240), ('Papel e Cartao', 'CA1100', 1100)]).
pontoRecolha( 15885 , -9.145088222 , 38.70790123 , '
    Misericordia' , 'Pc Sao Paulo' , [('Lixos', 'CV0240', 240)
    ]).
pontoRecolha( 15886 , -9.144888868 , 38.70762372 , '
    Misericordia' , 'Pc Sao Paulo' , [('Lixos', 'CV0140', 560),
     ('Lixos', 'CV0240', 240), ('Papel e Cartao', 'CV0140',
    140)]).
pontoRecolha( 15887 , -9.144475184 , 38.70763631 , '
    Misericordia' , 'Pc Sao Paulo' , [('Lixos', 'CV0240', 240)
    ]).
pontoRecolha( 15888 , -9.143298848 , 38.70655655 , '
    Misericordia' , 'Pc Duque da Terceira' , [('Lixos', 'CV0240
    ', 480), ('Papel e Cartao', 'CV0240', 240)]).
pontoRecolha( 15889 , -9.143404712 , 38.70672678 , '
    Misericordia' , 'Pc Duque da Terceira' , [('Lixos', 'CV0140
    ', 280), ('Lixos', 'CV0090', 90), ('Lixos', 'CV0120', 120),
     ('Papel e Cartao', 'CV0140', 280)]).
pontoRecolha( 15890 , -9.143611743 , 38.706734 , 'Misericordia
    ' , 'Pc Duque da Terceira' , [('Lixos', 'CV0090', 90), ('
    Lixos', 'CV0140', 560)]).
pontoRecolha( 15891 , -9.143759014 , 38.70657959 , '
```

```
    Misericordia' , 'Pc Duque da Terceira' , [('Lixos', 'CV0140
    ', 280), ('Papel e Cartao', 'CV0240', 240)]).
pontoRecolha( 15892 , -9.142005028 , 38.70692805 , '
    Misericordia' , 'Lg Corpo Santo' , [('Lixos', 'CV0140',
    140)]).
pontoRecolha( 15893 , -9.1420101 , 38.70728832 , 'Misericordia
    ' , 'Lg Corpo Santo' , [('Lixos', 'CV0090', 90), ('Lixos',
    'CV0140', 560), ('Lixos', 'CV0240', 480)]).
pontoRecolha( 15894 , -9.14208135 , 38.70744985 , '
    Misericordia' , 'Lg Corpo Santo' , [('Lixos', 'CV0240',
    240), ('Lixos', 'CV0140', 280), ('Papel e Cartao', 'CV0240
    ', 240)]).
pontoRecolha( 15896 , -9.149863686 , 38.70903075 , '
    Misericordia' , 'Bc Francisco Andre' , [('Lixos', 'CV0140',
     140)]).
pontoRecolha( 15897 , -9.14534815 , 38.7075837 , 'Misericordia
    ' , 'Tv de Sao Paulo' , [('Lixos', 'CV0140', 280), ('Lixos
    ', 'CV0240', 240), ('Papel e Cartao', 'CV0140', 140)]).
pontoRecolha( 15898 , -9.1443768 , 38.70636706 , 'Misericordia
    ' , 'Av 24 de Julho' , [('Lixos', 'CV0090', 90), ('Lixos',
    'CV0240', 240), ('Lixos', 'CV0140', 140)]).
pontoRecolha( 15899 , -9.144734149 , 38.70643602 , '
    Misericordia' , 'Av 24 de Julho' , [('Lixos', 'CV0140',
    140)]).
pontoRecolha( 19365 , -9.14751691 , 38.714609 , 'Misericordia'
     , 'R O Seculo' , [('Papel e Cartao', 'CV0240', 480), ('
    Vidro', 'CV0240', 240), ('Embalagens', 'CV0240', 480), ('
    Lixos', 'CV0240', 960)]).
pontoRecolha( 19216 , -9.151863405 , 38.71545391 , '
    Misericordia' , 'R Marcos Portugal' , [('Papel e Cartao', '
    CV0240', 240), ('Embalagens', 'CV0240', 240), ('Lixos', '
    CV0240', 240), ('Vidro', 'CV0240', 240)]).
pontoRecolha( 19280 , -9.147549562 , 38.71123078 , '
    Misericordia' , 'Tv Andre Valente' , [('Papel e Cartao', '
    CV0240', 720), ('Lixos', 'CV0240', 2160), ('Embalagens', '
    CV0240', 480), ('Vidro', 'CV0240', 240)]).
```

```
pontoRecolha( 19301 , -9.144224007 , 38.70694488 , '
   Misericordia' , 'R Remolares' , [('Vidro', 'VSP', 1500)]).
pontoRecolha( 19315 , -9.144407318 , 38.71015909 , '
   Misericordia' , 'R Emenda' , [('Vidro', 'CV0140', 140)]).
pontoRecolha( 19300 , -9.144900544 , 38.71089346 , '
   Misericordia' , 'R Salgadeiras' , [('Lixos', 'CV0240', 480)
   ]).
pontoRecolha( 19407 , -9.143880634 , 38.70949707 , '
   Misericordia' , 'Tv Guilherme Cossoul' , [('Papel e Cartao
   ', 'CV0240', 240), ('Embalagens', 'CV0240', 240)]).
pontoRecolha( 19282 , -9.151388551 , 38.70873822 , '
   Misericordia' , 'Lg Conde Barao' , [('Papel e Cartao', '
   CV0140', 140), ('Embalagens', 'CV0140', 140)]).
pontoRecolha( 19257 , -9.153108561 , 38.71327215 , '
   Misericordia' , 'R Sao Bento' , [('Vidro', 'CV0140', 280)]
   .
pontoRecolha( 19293 , -9.15311831 , 38.71637976 , '
   Misericordia' , 'R Imprensa Nacional' , [('Vidro', 'CV0140
   ', 140)]).
pontoRecolha( 19310 , -9.142260747 , 38.70794373 , '
   Misericordia' , 'R Vitor Cordon' , [('Vidro', 'CV0240',
   240)]).
pontoRecolha( 19319 , -9.144813491 , 38.70798468 , '
   Misericordia' , 'R Sao Paulo' , [('Embalagens', 'CV0140',
   140)]).
pontoRecolha( 19236 , -9.153608439 , 38.71446582 , '
   Misericordia' , 'R Sao Bento' , [('Vidro', 'CV0140', 140),
   ('Organicos', 'CV0140', 140)]).
pontoRecolha( 19295 , -9.144603489 , 38.71428298 , '
   Misericordia' , 'R Teixeira' , [('Vidro', 'CV0140', 140)]).
pontoRecolha( 19371 , -9.14464752 , 38.71088664 , '
   Misericordia' , 'R Salgadeiras' , [('Lixos', 'CV0240', 720)
   ]).
pontoRecolha( 19390 , -9.152425179 , 38.709684 , 'Misericordia
   ' , 'R Posso dos Negros' , [('Lixos', 'CV0240', 720)]).
pontoRecolha( 19278 , -9.153055613 , 38.71197549 , '
```

```
    Misericordia' , 'R Correia Garasao' , [('Vidro', 'CV0240',
    240), ('Organicos', 'CV0240', 240)]).
pontoRecolha( 19415 , -9.143011665 , 38.70738775 , '
    Misericordia' , 'R Sao Paulo' , [('Embalagens', 'CV0140',
    140)]).
pontoRecolha( 19337 , -9.143856881 , 38.70944323 , '
    Misericordia' , 'Tv Guilherme Cossoul' , [('Vidro', 'CV0140
    ', 140), ('Papel e Cartao', 'CV0140', 140), ('Embalagens',
    'CV0140', 140)]).
pontoRecolha( 21843 , -9.15264388 , 38.71051081 , '
    Misericordia' , 'R Sao Bento' , [('Organicos', 'CV0090',
    90), ('Vidro', 'CV0090', 90)]).
pontoRecolha( 21844 , -9.151300393 , 38.71224303 , '
    Misericordia' , 'R Cruz dos Poiais de Sao Bento' , [('Papel
     e Cartao', 'CV0240', 240), ('Embalagens', 'CV0240', 240),
    ('Lixos', 'CV0240', 240), ('Vidro', 'CV0240', 240)]).
pontoRecolha( 21957 , -9.147528628 , 38.70569115 , '
    Misericordia' , 'R da Cintura do Porto de Lisboa' , [('
    Embalagens', 'CV0240', 480), ('Lixos', 'CV0240', 240), ('
    Vidro', 'CV0240', 240)]).
pontoRecolha( 21959 , -9.15077561 , 38.70848234 , '
    Misericordia' , 'R Instituto Industrial' , [('Lixos', '
    CV0240', 240), ('Embalagens', 'CV0240', 480), ('Vidro', '
    CV0140', 560), ('Organicos', 'CV0240', 720)]).
pontoRecolha( 21911 , -9.146054168 , 38.70791988 , '
    Misericordia' , 'Tv Carvalho' , [('Papel e Cartao', '
    ECIS3000', 6000), ('Embalagens', 'ECIS3000', 6000), ('Vidro
    ', 'ECIS3000', 3000), ('Lixos', 'ECIS3000', 6000)]).
pontoRecolha( 21952 , -9.149067977 , 38.70885752 , '
    Misericordia' , 'R da Boavista' , [('Papel e Cartao', '
    CV0090', 90), ('Organicos', 'CV0090', 90), ('Vidro', '
    CV0140', 140), ('Embalagens', 'CV0090', 90)]).
pontoRecolha( 21969 , -9.15158074 , 38.70770064 , '
    Misericordia' , 'R Dom Luis I' , [('Papel e Cartao', '
    CV0240', 240), ('Embalagens', 'CV0090', 90)]).
pontoRecolha( 21854 , -9.144441604 , 38.70933007 , '
```

```
    Misericordia' , 'R Emenda' , [('Papel e Cartao', 'CV0140',
    140), ('Vidro', 'CV0240', 240), ('Embalagens', 'CV0140',
    140)]).
pontoRecolha( 21938 , -9.144083429 , 38.71164819 , '
    Misericordia' , 'R Diario de Noticias' , [('Organicos', '
    CV0240', 240)]).
pontoRecolha( 21961 , -9.143292489 , 38.70773663 , '
    Misericordia' , 'R do Alecrim' , [('Embalagens', 'CV0090',
    90)]).
pontoRecolha( 21866 , -9.147315299 , 38.71173725 , '
    Misericordia' , 'R O Seculo' , [('Papel e Cartao', 'CV0090
    ', 90), ('Organicos', 'CV0090', 90), ('Vidro', 'CV0090',
    90), ('Embalagens', 'CV0090', 90), ('Lixos', 'CV0090', 90)
    ]).
pontoRecolha( 21966 , -9.146424706 , 38.71054695 , '
    Misericordia' , 'R Marechal Saldanha' , [('Lixos', 'CV0140
    ', 420), ('Organicos', 'CV0140', 140), ('Vidro', 'CV0140',
    140)]).
pontoRecolha( 21967 , -9.144977831 , 38.71066759 , '
    Misericordia' , 'R Loreto' , [('Organicos', 'CV0140', 140)
    ]).
pontoRecolha( 21944 , -9.143233746 , 38.70764706 , '
    Misericordia' , 'R Ferragial' , [('Vidro', 'CV0240', 240),
    ('Organicos', 'CV0240', 240)]).
pontoRecolha( 21932 , -9.148416742 , 38.70752102 , '
    Misericordia' , 'R Dom Luis I' , [('Embalagens', 'CV0240',
    240), ('Papel e Cartao', 'CV0240', 240)]).
pontoRecolha( 21889 , -9.146202328 , 38.70782851 , '
    Misericordia' , 'Tv Carvalho' , [('Vidro', 'CV0090', 90)]).
pontoRecolha( 21925 , -9.142597828 , 38.70902175 , '
    Misericordia' , 'R Antonio Maria Cardoso' , [('Papel e
    Cartao', 'CV0090', 90), ('Lixos', 'CV0140', 140)]).
pontoRecolha( 21927 , -9.145845743 , 38.71025471 , '
    Misericordia' , 'Tv Sequeiro' , [('Lixos', 'CV0240', 1200)
    ]).
pontoRecolha( 21939 , -9.144084572 , 38.71172925 , '
```

```
    Misericordia' , 'R Diario de Noticias' , [('Organicos', '
    CV0240', 240)]).
pontoRecolha( 21949 , -9.152256635 , 38.71400022 , '
    Misericordia' , 'R Quintinha' , [('Organicos', 'CV0140',
    140), ('Vidro', 'CV0140', 140)]).
```

A.3.2 Lista de Adjacências

```
:- module(arestas, [aresta/3]).
aresta( 15805 , 15808 , 0.1175709600401816 ).
aresta( 15805 , 15865 , 0.1301916442954011 ).
aresta( 15805 , 15866 , 0.10645682662066398 ).
aresta( 15805 , 15867 , 0.09972957970367313 ).
aresta( 15805 , 15888 , 0.1673196539492436 ).
aresta( 15805 , 15889 , 0.14898335805463714 ).
aresta( 15805 , 15890 , 0.15156149079737344 ).
aresta( 15805 , 15891 , 0.17214242441866054 ).
aresta( 15806 , 15808 , 0.12308119095685881 ).
aresta( 15806 , 15865 , 0.1238785651929606 ).
aresta( 15806 , 15866 , 0.1025961874905506 ).
aresta( 15806 , 15867 , 0.10093126885579122 ).
aresta( 15806 , 15888 , 0.1674777400224743 ).
aresta( 15806 , 15889 , 0.14856919004515645 ).
aresta( 15806 , 15890 , 0.1499947129450299 ).
aresta( 15806 , 15891 , 0.1700288249003034 ).
aresta( 15807 , 15808 , 0.10299068798547002 ).
aresta( 15807 , 15865 , 0.09014670342548893 ).
aresta( 15807 , 15866 , 0.037173644504773916 ).
aresta( 15807 , 15867 , 0.034746620782516384 ).
aresta( 15807 , 15888 , 0.08446362231690586 ).
aresta( 15807 , 15889 , 0.06386484848295128 ).
aresta( 15807 , 15890 , 0.06413044736620271 ).
aresta( 15807 , 15891 , 0.08518044310516185 ).
aresta( 15808 , 15805 , 0.1175709600401816 ).
aresta( 15808 , 15806 , 0.12308119095685881 ).
aresta( 15808 , 15807 , 0.10299068798547002 ).
```

```
aresta( 15808 , 15809 , 0.03592933002302158 ).
aresta( 15808 , 15810 , 0.06982398499527023 ).
aresta( 15808 , 15855 , 0.10807873170235617 ).
aresta( 15808 , 15856 , 0.18299869650530085 ).
aresta( 15808 , 15857 , 0.18572121157978277 ).
aresta( 15808 , 15858 , 0.1083717645802594 ).
aresta( 15808 , 15859 , 0.2515664956322809 ).
aresta( 15808 , 15860 , 0.34146764498964194 ).
aresta( 15808 , 15861 , 0.4038225914628467 ).
aresta( 15808 , 15862 , 0.4371645902993397 ).
aresta( 15808 , 15863 , 0.4705886448279587 ).
aresta( 15808 , 15864 , 0.5033950769721541 ).
aresta( 15808 , 15892 , 0.07473525468785382 ).
aresta( 15808 , 15893 , 0.059987607279522556 ).
aresta( 15808 , 15894 , 0.05362319943152387 ).
aresta( 15809 , 15808 , 0.03592933002302158 ).
aresta( 15809 , 15811 , 0.04069099001040787 ).
aresta( 15809 , 15812 , 0.040175925385031876 ).
aresta( 15809 , 15876 , 0.19770728008010485 ).
aresta( 15809 , 15877 , 0.14963777685809218 ).
aresta( 15809 , 15878 , 0.07251894481249586 ).
aresta( 15809 , 15879 , 0.08219095249619063 ).
aresta( 15810 , 15808 , 0.06982398499527023 ).
aresta( 15810 , 15811 , 0.04052406950301554 ).
aresta( 15810 , 15812 , 0.050396532946324187 ).
aresta( 15810 , 15876 , 0.18421706529249796 ).
aresta( 15810 , 15877 , 0.11928150171583017 ).
aresta( 15810 , 15878 , 0.036636644451277005 ).
aresta( 15810 , 15879 , 0.045785440164822215 ).
aresta( 15811 , 15809 , 0.04069099001040787 ).
aresta( 15811 , 15810 , 0.04052406950301554 ).
aresta( 15811 , 15813 , 0.9973819661455475 ).
aresta( 15811 , 15814 , 1.0530953975006754 ).
aresta( 15811 , 15815 , 1.077586465899776 ).
aresta( 15811 , 15888 , 0.10972759086694633 ).
aresta( 15811 , 15889 , 0.11412108558758242 ).
```

```
aresta( 15811 , 15890 , 0.13619181012302536 ).
aresta( 15811 , 15891 , 0.15626047428911907 ).
aresta( 15811 , 15892 , 0.0452414012283684 ).
aresta( 15811 , 15893 , 0.05448062104408998 ).
aresta( 15811 , 15894 , 0.061560407978865146 ).
aresta( 15812 , 15809 , 0.040175925385031876 ).
aresta( 15812 , 15810 , 0.050396532946324187 ).
aresta( 15812 , 15813 , 0.9312596442952551 ).
aresta( 15812 , 15814 , 0.9861822160033922 ).
aresta( 15812 , 15815 , 1.0104630302543351 ).
aresta( 15812 , 15888 , 0.042890726135212186 ).
aresta( 15812 , 15889 , 0.041706195959626 ).
aresta( 15812 , 15890 , 0.06354301556764921 ).
aresta( 15812 , 15891 , 0.08408062018952182 ).
aresta( 15812 , 15892 , 0.11586472587586745 ).
aresta( 15812 , 15893 , 0.12446235599839334 ).
aresta( 15812 , 15894 , 0.12546982148339292 ).
aresta( 15813 , 15811 , 0.9973819661455475 ).
aresta( 15813 , 15812 , 0.9312596442952551 ).
aresta( 15813 , 15818 , 0.23955676030661835 ).
aresta( 15813 , 15819 , 0.39899931208949063 ).
aresta( 15813 , 15820 , 0.2963901640645702 ).
aresta( 15813 , 15821 , 0.23648546627490813 ).
aresta( 15813 , 15822 , 0.19468070330355194 ).
aresta( 15813 , 15823 , 0.11897385393121374 ).
aresta( 15813 , 15824 , 0.05263624292551712 ).
aresta( 15813 , 15830 , 0.052205351183031774 ).
aresta( 15813 , 15831 , 0.0944296101403991 ).
aresta( 15814 , 15811 , 1.0530953975006754 ).
aresta( 15814 , 15812 , 0.9861822160033922 ).
aresta( 15814 , 15818 , 0.30072370796857 ).
aresta( 15814 , 15819 , 0.45792937236447157 ).
aresta( 15814 , 15820 , 0.35602424745254885 ).
aresta( 15814 , 15821 , 0.29667511039273725 ).
aresta( 15814 , 15822 , 0.2552575955349181 ).
aresta( 15814 , 15823 , 0.1801391683941532 ).
```

```
aresta( 15814 , 15824 , 0.11412199287323468 ).
aresta( 15814 , 15830 , 0.03719174873725381 ).
aresta( 15814 , 15831 , 0.08957710715638884 ).
aresta( 15815 , 15811 , 1.077586465899776 ).
aresta( 15815 , 15812 , 1.0104630302543351 ).
aresta( 15815 , 15818 , 0.3262932016753131 ).
aresta( 15815 , 15819 , 0.48326894692243694 ).
aresta( 15815 , 15820 , 0.381496834116299 ).
aresta( 15815 , 15821 , 0.3222223933366646 ).
aresta( 15815 , 15822 , 0.28083589328706854 ).
aresta( 15815 , 15823 , 0.20570225350781032 ).
aresta( 15815 , 15824 , 0.13950770906025442 ).
aresta( 15815 , 15830 , 0.05700217068080114 ).
aresta( 15815 , 15831 , 0.10214439587281697 ).
aresta( 15818 , 15813 , 0.23955676030661835 ).
aresta( 15818 , 15814 , 0.30072370796857 ).
aresta( 15818 , 15815 , 0.3262932016753131 ).
aresta( 15818 , 15819 , 0.17048232166804714 ).
aresta( 15818 , 15820 , 0.07026209160945704 ).
aresta( 15818 , 15821 , 0.027742573233407564 ).
aresta( 15818 , 15822 , 0.04829613568149859 ).
aresta( 15818 , 15823 , 0.1205990810319327 ).
aresta( 15818 , 15824 , 0.18756074905844386 ).
aresta( 15818 , 21952 , 0.02408648236961493 ).
aresta( 15819 , 15813 , 0.39899931208949063 ).
aresta( 15819 , 15814 , 0.45792937236447157 ).
aresta( 15819 , 15815 , 0.48326894692243694 ).
aresta( 15819 , 15818 , 0.17048232166804714 ).
aresta( 15819 , 15825 , 0.48880244684054125 ).
aresta( 15819 , 15834 , 0.3666816677478945 ).
aresta( 15819 , 15836 , 0.3292890132767965 ).
aresta( 15819 , 15855 , 0.4720326243013291 ).
aresta( 15819 , 15856 , 0.39706271012411476 ).
aresta( 15819 , 15857 , 0.39434488077375573 ).
aresta( 15819 , 15858 , 0.472091176767095 ).
aresta( 15819 , 15859 , 0.3285175000764331 ).
```

```
aresta( 15819 , 15860 , 0.23897808732370565 ).
aresta( 15819 , 15861 , 0.17747567033198866 ).
aresta( 15819 , 15862 , 0.14462833972920425 ).
aresta( 15819 , 15863 , 0.11111017903365746 ).
aresta( 15819 , 15864 , 0.07711857375903375 ).
aresta( 15819 , 15881 , 0.21728638852314078 ).
aresta( 15819 , 15882 , 0.33030776610379786 ).
aresta( 15819 , 15896 , 0.25019095789108836 ).
aresta( 15819 , 19282 , 0.41396420118927557 ).
aresta( 15819 , 21959 , 0.34584157941968247 ).
aresta( 15820 , 15813 , 0.2963901640645702 ).
aresta( 15820 , 15814 , 0.35602424745254885 ).
aresta( 15820 , 15815 , 0.381496834116299 ).
aresta( 15820 , 15818 , 0.07026209160945704 ).
aresta( 15820 , 15825 , 0.38892824777668894 ).
aresta( 15820 , 15834 , 0.29771514665727605 ).
aresta( 15820 , 15836 , 0.23759366632524226 ).
aresta( 15820 , 15855 , 0.5753687571226797 ).
aresta( 15820 , 15856 , 0.5004734639164371 ).
aresta( 15820 , 15857 , 0.4976941652678774 ).
aresta( 15820 , 15858 , 0.575358485993852 ).
aresta( 15820 , 15859 , 0.43195476419185413 ).
aresta( 15820 , 15860 , 0.34250874119523045 ).
aresta( 15820 , 15861 , 0.28097235879952154 ).
aresta( 15820 , 15862 , 0.24800961034295688 ).
aresta( 15820 , 15863 , 0.21441791357772708 ).
aresta( 15820 , 15864 , 0.18062318201018374 ).
aresta( 15820 , 15881 , 0.1140099248192527 ).
aresta( 15820 , 15882 , 0.22678803874064105 ).
aresta( 15820 , 15896 , 0.14674335695660054 ).
aresta( 15820 , 19282 , 0.31154408173303305 ).
aresta( 15820 , 21959 , 0.24522429471152277 ).
aresta( 15821 , 15813 , 0.23648546627490813 ).
aresta( 15821 , 15814 , 0.29667511039273725 ).
aresta( 15821 , 15815 , 0.3222223933366646 ).
aresta( 15821 , 15818 , 0.027742573233407564 ).
```

```
aresta( 15821 , 15825 , 0.33127737102585436 ).
aresta( 15821 , 15834 , 0.2676615341007406 ).
aresta( 15821 , 15836 , 0.18941657159247882 ).
aresta( 15821 , 15855 , 0.6360718804661127 ).
aresta( 15821 , 15856 , 0.5612135923669008 ).
aresta( 15821 , 15857 , 0.5584075935459715 ).
aresta( 15821 , 15858 , 0.6360303535951343 ).
aresta( 15821 , 15859 , 0.4927092891263625 ).
aresta( 15821 , 15860 , 0.4033040859191861 ).
aresta( 15821 , 15861 , 0.3417625030310438 ).
aresta( 15821 , 15862 , 0.30877172596043106 ).
aresta( 15821 , 15863 , 0.27516928146405945 ).
aresta( 15821 , 15864 , 0.24141913959830968 ).
aresta( 15821 , 15881 , 0.05387502584316203 ).
aresta( 15821 , 15882 , 0.1660076699219572 ).
aresta( 15821 , 15896 , 0.08611844636089686 ).
aresta( 15821 , 19282 , 0.251796820422542 ).
aresta( 15821 , 21959 , 0.18757194958837162 ).
aresta( 15822 , 15813 , 0.19468070330355194 ).
aresta( 15822 , 15814 , 0.2552575955349181 ).
aresta( 15822 , 15815 , 0.28083589328706854 ).
aresta( 15822 , 15818 , 0.04829613568149859 ).
aresta( 15822 , 15825 , 0.2912034304561153 ).
aresta( 15822 , 15834 , 0.2517045887098845 ).
aresta( 15822 , 15836 , 0.15963434914251048 ).
aresta( 15822 , 15855 , 0.6783060752581498 ).
aresta( 15822 , 15856 , 0.6034896802091864 ).
aresta( 15822 , 15857 , 0.6006552163514353 ).
aresta( 15822 , 15858 , 0.6782334621842718 ).
aresta( 15822 , 15859 , 0.5350040651998803 ).
aresta( 15822 , 15860 , 0.4456599835002005 ).
aresta( 15822 , 15861 , 0.38416111051560636 ).
aresta( 15822 , 15862 , 0.35117878090543064 ).
aresta( 15822 , 15863 , 0.317577112305976 ).
aresta( 15822 , 15864 , 0.28381266595762344 ).
aresta( 15822 , 15881 , 0.016733552871422495 ).
```

```
aresta( 15822 , 15882 , 0.12360193206695314 ).
aresta( 15822 , 15896 , 0.044509042298095124 ).
aresta( 15822 , 19282 , 0.21010587439889225 ).
aresta( 15822 , 21959 , 0.1481206104511038 ).
aresta( 15823 , 15813 , 0.11897385393121374 ).
aresta( 15823 , 15814 , 0.1801391683941532 ).
aresta( 15823 , 15815 , 0.20570225350781032 ).
aresta( 15823 , 15818 , 0.1205990810319327 ).
aresta( 15823 , 15825 , 0.2187498374533051 ).
aresta( 15823 , 15834 , 0.23400923516087904 ).
aresta( 15823 , 15836 , 0.12019023463503108 ).
aresta( 15823 , 15855 , 0.753862848718233 ).
aresta( 15823 , 15856 , 0.6791816986900128 ).
aresta( 15823 , 15857 , 0.6762655279634507 ).
aresta( 15823 , 15858 , 0.7537040712583678 ).
aresta( 15823 , 15859 , 0.610766119711693 ).
aresta( 15823 , 15860 , 0.5216404706896802 ).
aresta( 15823 , 15861 , 0.46034256949224667 ).
aresta( 15823 , 15862 , 0.42743928437367085 ).
aresta( 15823 , 15863 , 0.3938576199048924 ).
aresta( 15823 , 15864 , 0.3599854485118193 ).
aresta( 15823 , 15881 , 0.06931176597330761 ).
aresta( 15823 , 15882 , 0.0478983535936495 ).
aresta( 15823 , 15896 , 0.03778678611000041 ).
aresta( 15823 , 19282 , 0.13458341209745994 ).
aresta( 15823 , 21959 , 0.08063240937530029 ).
aresta( 15824 , 15813 , 0.05263624292551712 ).
aresta( 15824 , 15814 , 0.11412199287323468 ).
aresta( 15824 , 15815 , 0.13950770906025442 ).
aresta( 15824 , 15818 , 0.18756074905844386 ).
aresta( 15824 , 15825 , 0.15601986433867984 ).
aresta( 15824 , 15834 , 0.23420101457767942 ).
aresta( 15824 , 15836 , 0.11583374461917313 ).
aresta( 15824 , 15855 , 0.8198653929643684 ).
aresta( 15824 , 15856 , 0.7453487376782704 ).
aresta( 15824 , 15857 , 0.7423482502860496 ).
```

```
aresta( 15824 , 15858 , 0.8196171241097907 ).
aresta( 15824 , 15859 , 0.6770294530105255 ).
aresta( 15824 , 15860 , 0.5881672156384835 ).
aresta( 15824 , 15861 , 0.5271137275313654 ).
aresta( 15824 , 15862 , 0.4943144864730252 ).
aresta( 15824 , 15863 , 0.46076757801448864 ).
aresta( 15824 , 15864 , 0.42678661205119306 ).
aresta( 15824 , 15881 , 0.13634615979374337 ).
aresta( 15824 , 15882 , 0.02561104196816848 ).
aresta( 15824 , 15896 , 0.10371423546551421 ).
aresta( 15824 , 19282 , 0.06837628560243393 ).
aresta( 15824 , 21959 , 0.04655616646221061 ).
aresta( 15825 , 15819 , 0.48880244684054125 ).
aresta( 15825 , 15820 , 0.38892824777668894 ).
aresta( 15825 , 15821 , 0.33127737102585436 ).
aresta( 15825 , 15822 , 0.2912034304561153 ).
aresta( 15825 , 15823 , 0.2187498374533051 ).
aresta( 15825 , 15824 , 0.15601986433867984 ).
aresta( 15825 , 15827 , 0.0483499028107009 ).
aresta( 15825 , 15828 , 0.031328825478549496 ).
aresta( 15827 , 15825 , 0.0483499028107009 ).
aresta( 15827 , 15830 , 0.10675547797738942 ).
aresta( 15827 , 15831 , 0.12198212229218193 ).
aresta( 15828 , 15825 , 0.031328825478549496 ).
aresta( 15828 , 15830 , 0.03315105287251279 ).
aresta( 15828 , 15831 , 0.05433184817041891 ).
aresta( 15830 , 15813 , 0.052205351183031774 ).
aresta( 15830 , 15814 , 0.03719174873725381 ).
aresta( 15830 , 15815 , 0.05700217068080114 ).
aresta( 15830 , 15827 , 0.10675547797738942 ).
aresta( 15830 , 15828 , 0.03315105287251279 ).
aresta( 15830 , 15832 , 0.4939831924758537 ).
aresta( 15830 , 15845 , 0.0989286381646894 ).
aresta( 15830 , 15846 , 0.11045584119791761 ).
aresta( 15830 , 15847 , 0.19791614192946327 ).
aresta( 15830 , 15848 , 0.3722218913138464 ).
```

```
aresta( 15830 , 21969 , 0.077361732777701 ).
aresta( 15830 , 21932 , 0.35930582938194205 ).
aresta( 15831 , 15813 , 0.0944296101403991 ).
aresta( 15831 , 15814 , 0.08957710715638884 ).
aresta( 15831 , 15815 , 0.10214439587281697 ).
aresta( 15831 , 15827 , 0.12198212229218193 ).
aresta( 15831 , 15828 , 0.05433184817041891 ).
aresta( 15831 , 15832 , 0.5039002921544257 ).
aresta( 15831 , 15845 , 0.07472068953340243 ).
aresta( 15831 , 15846 , 0.06122771128318302 ).
aresta( 15831 , 15847 , 0.16430864443761486 ).
aresta( 15831 , 15848 , 0.3452782743070767 ).
aresta( 15831 , 21969 , 0.032767950353301496 ).
aresta( 15831 , 21932 , 0.3317091939197365 ).
aresta( 15832 , 15830 , 0.4939831924758537 ).
aresta( 15832 , 15831 , 0.5039002921544257 ).
aresta( 15832 , 15833 , 0.12425394741016477 ).
aresta( 15833 , 15832 , 0.12425394741016477 ).
aresta( 15833 , 15834 , 0.40418251555271784 ).
aresta( 15833 , 15836 , 0.315360537029306 ).
aresta( 15834 , 15819 , 0.3666816677478945 ).
aresta( 15834 , 15820 , 0.29771514665727605 ).
aresta( 15834 , 15821 , 0.2676615341007406 ).
aresta( 15834 , 15822 , 0.2517045887098845 ).
aresta( 15834 , 15823 , 0.23400923516087904 ).
aresta( 15834 , 15824 , 0.23420101457767942 ).
aresta( 15834 , 15833 , 0.40418251555271784 ).
aresta( 15834 , 15838 , 0.3944725660822953 ).
aresta( 15834 , 15845 , 0.19187530093769126 ).
aresta( 15834 , 15846 , 0.10284306788105266 ).
aresta( 15834 , 15847 , 0.1070065878200047 ).
aresta( 15834 , 15848 , 0.2537938739192756 ).
aresta( 15834 , 15898 , 0.6749790391109016 ).
aresta( 15834 , 15899 , 0.6350368491880308 ).
aresta( 15834 , 21952 , 0.27379692159698965 ).
aresta( 15834 , 21969 , 0.15821868148902074 ).
```

```
aresta( 15834 , 21932 , 0.23991222282629562 ).
aresta( 15836 , 15819 , 0.3292890132767965 ).
aresta( 15836 , 15820 , 0.23759366632524226 ).
aresta( 15836 , 15821 , 0.18941657159247882 ).
aresta( 15836 , 15822 , 0.15963434914251048 ).
aresta( 15836 , 15823 , 0.12019023463503108 ).
aresta( 15836 , 15824 , 0.11583374461917313 ).
aresta( 15836 , 15833 , 0.315360537029306 ).
aresta( 15836 , 15838 , 0.308725968378575 ).
aresta( 15836 , 15845 , 0.16226237362539667 ).
aresta( 15836 , 15846 , 0.05231015642635353 ).
aresta( 15836 , 15847 , 0.07379217851586012 ).
aresta( 15836 , 15848 , 0.2525404971700774 ).
aresta( 15836 , 15898 , 0.7026036594983183 ).
aresta( 15836 , 15899 , 0.6624122852372273 ).
aresta( 15836 , 21952 , 0.19581610553138332 ).
aresta( 15836 , 21969 , 0.11585726005593713 ).
aresta( 15836 , 21932 , 0.23908570935167112 ).
aresta( 15838 , 15834 , 0.3944725660822953 ).
aresta( 15838 , 15836 , 0.308725968378575 ).
aresta( 15838 , 15841 , 0.2553971763692401 ).
aresta( 15841 , 15838 , 0.2553971763692401 ).
aresta( 15841 , 15842 , 0.10285340300137266 ).
aresta( 15841 , 15843 , 0.15600597361217333 ).
aresta( 15841 , 15844 , 0.1178256689603684 ).
aresta( 15841 , 15855 , 0.39483878522466226 ).
aresta( 15841 , 15856 , 0.32107939130181684 ).
aresta( 15841 , 15857 , 0.31775278204456264 ).
aresta( 15841 , 15858 , 0.39436075725389214 ).
aresta( 15841 , 15859 , 0.25359293330711147 ).
aresta( 15841 , 15860 , 0.1682461769864437 ).
aresta( 15841 , 15861 , 0.11451511181979397 ).
aresta( 15841 , 15862 , 0.0892251278790514 ).
aresta( 15841 , 15863 , 0.06711528483508904 ).
aresta( 15841 , 15864 , 0.05048065729409898 ).
aresta( 15842 , 15841 , 0.10285340300137266 ).
```

```
aresta( 15842 , 15845 , 0.6471832308411296 ).
aresta( 15842 , 15846 , 0.5302970455145156 ).
aresta( 15842 , 15847 , 0.4197403986417306 ).
aresta( 15842 , 15848 , 0.23889803255464104 ).
aresta( 15842 , 15852 , 0.14522543385036255 ).
aresta( 15842 , 15853 , 0.0403951048377961 ).
aresta( 15842 , 15855 , 0.29293394757378866 ).
aresta( 15842 , 15856 , 0.22009994072900727 ).
aresta( 15842 , 15857 , 0.2164727101521401 ).
aresta( 15842 , 15858 , 0.29225928238893584 ).
aresta( 15842 , 15859 , 0.15416361382075708 ).
aresta( 15842 , 15860 , 0.07906637567176235 ).
aresta( 15842 , 15861 , 0.06216402071524202 ).
aresta( 15842 , 15862 , 0.0746049354829072 ).
aresta( 15842 , 15863 , 0.09486691304906979 ).
aresta( 15842 , 15864 , 0.11750903479629167 ).
aresta( 15842 , 15885 , 0.11750447418894308 ).
aresta( 15842 , 15886 , 0.13995754104762032 ).
aresta( 15842 , 15887 , 0.185311234178214 ).
aresta( 15842 , 19319 , 0.14882548825152334 ).
aresta( 15842 , 19415 , 0.34919412869573424 ).
aresta( 15842 , 21911 , 0.017953208396719114 ).
aresta( 15842 , 21889 , 0.008078567212858094 ).
aresta( 15843 , 15841 , 0.15600597361217333 ).
aresta( 15843 , 15845 , 0.7033237015996293 ).
aresta( 15843 , 15846 , 0.5865349444626715 ).
aresta( 15843 , 15847 , 0.4760547533675795 ).
aresta( 15843 , 15848 , 0.2956492188321821 ).
aresta( 15843 , 15852 , 0.10298461127008156 ).
aresta( 15843 , 15853 , 0.04294517101494889 ).
aresta( 15843 , 15855 , 0.2390850818696364 ).
aresta( 15843 , 15856 , 0.16507509265656722 ).
aresta( 15843 , 15857 , 0.16179091345760543 ).
aresta( 15843 , 15858 , 0.23877540577370038 ).
aresta( 15843 , 15859 , 0.09789797340501079 ).
aresta( 15843 , 15860 , 0.031335564883998135 ).
```

```
aresta( 15843 , 15861 , 0.0683397692679218 ).
aresta( 15843 , 15862 , 0.09856350339809834 ).
aresta( 15843 , 15863 , 0.12929589401358457 ).
aresta( 15843 , 15864 , 0.15932215692885204 ).
aresta( 15843 , 15885 , 0.061029835492814634 ).
aresta( 15843 , 15886 , 0.08797262490316851 ).
aresta( 15843 , 15887 , 0.13173863034473365 ).
aresta( 15843 , 19319 , 0.09201970889419547 ).
aresta( 15843 , 19415 , 0.29582623142849684 ).
aresta( 15843 , 21911 , 0.04596337836530813 ).
aresta( 15843 , 21889 , 0.06254799163777187 ).
aresta( 15844 , 15841 , 0.1178256689603684 ).
aresta( 15844 , 15845 , 0.6667888501510649 ).
aresta( 15844 , 15846 , 0.5505566872664945 ).
aresta( 15844 , 15847 , 0.4406734925587073 ).
aresta( 15844 , 15848 , 0.26366026464950393 ).
aresta( 15844 , 15852 , 0.15205270842591265 ).
aresta( 15844 , 15853 , 0.06786961771425697 ).
aresta( 15844 , 15855 , 0.283135058264965 ).
aresta( 15844 , 15856 , 0.2081563147881344 ).
aresta( 15844 , 15857 , 0.20544592305846868 ).
aresta( 15844 , 15858 , 0.28324195794353657 ).
aresta( 15844 , 15859 , 0.13962688444722474 ).
aresta( 15844 , 15860 , 0.05090574486761903 ).
aresta( 15844 , 15861 , 0.0212590249747395 ).
aresta( 15844 , 15862 , 0.049484839846525304 ).
aresta( 15844 , 15863 , 0.08093960779329344 ).
aresta( 15844 , 15864 , 0.11246369988585665 ).
aresta( 15844 , 15885 , 0.10368105406846645 ).
aresta( 15844 , 15886 , 0.1353424531786286 ).
aresta( 15844 , 15887 , 0.1771427822486683 ).
aresta( 15844 , 19319 , 0.13145411432242124 ).
aresta( 15844 , 19415 , 0.34012548312537816 ).
aresta( 15844 , 21911 , 0.02775770717175955 ).
aresta( 15844 , 21889 , 0.043767652984213354 ).
aresta( 15845 , 15830 , 0.0989286381646894 ).
```

```
aresta( 15845 , 15831 , 0.07472068953340243 ).
aresta( 15845 , 15834 , 0.19187530093769126 ).
aresta( 15845 , 15836 , 0.16226237362539667 ).
aresta( 15845 , 15842 , 0.6471832308411296 ).
aresta( 15845 , 15843 , 0.7033237015996293 ).
aresta( 15845 , 15844 , 0.6667888501510649 ).
aresta( 15845 , 15849 , 0.5392253151616835 ).
aresta( 15845 , 15851 , 0.5837577247647936 ).
aresta( 15845 , 21959 , 0.16439382208225722 ).
aresta( 15846 , 15830 , 0.11045584119791761 ).
aresta( 15846 , 15831 , 0.06122771128318302 ).
aresta( 15846 , 15834 , 0.10284306788105266 ).
aresta( 15846 , 15836 , 0.05231015642635353 ).
aresta( 15846 , 15842 , 0.5302970455145156 ).
aresta( 15846 , 15843 , 0.5865349444626715 ).
aresta( 15846 , 15844 , 0.5505566872664945 ).
aresta( 15846 , 15849 , 0.42281459146367467 ).
aresta( 15846 , 15851 , 0.4667368555664502 ).
aresta( 15846 , 21959 , 0.09780519193824547 ).
aresta( 15847 , 15830 , 0.19791614192946327 ).
aresta( 15847 , 15831 , 0.16430864443761486 ).
aresta( 15847 , 15834 , 0.107006587820047 ).
aresta( 15847 , 15836 , 0.07379217851586012 ).
aresta( 15847 , 15842 , 0.4197403986417306 ).
aresta( 15847 , 15843 , 0.4760547533675795 ).
aresta( 15847 , 15844 , 0.4406734925587073 ).
aresta( 15847 , 15849 , 0.31368580359198933 ).
aresta( 15847 , 15851 , 0.35615194144645196 ).
aresta( 15847 , 21959 , 0.133046278514393 ).
aresta( 15848 , 15830 , 0.3722218913138464 ).
aresta( 15848 , 15831 , 0.3452782743070767 ).
aresta( 15848 , 15834 , 0.2537938739192756 ).
aresta( 15848 , 15836 , 0.2525404971700774 ).
aresta( 15848 , 15842 , 0.23889803255464104 ).
aresta( 15848 , 15843 , 0.2956492188321821 ).
aresta( 15848 , 15844 , 0.26366026464950393 ).
```

```
aresta( 15848 , 15849 , 0.13551063745642336 ).
aresta( 15848 , 15851 , 0.17417827864984836 ).
aresta( 15848 , 21959 , 0.2937153532289583 ).
aresta( 15849 , 15845 , 0.5392253151616835 ).
aresta( 15849 , 15846 , 0.42281459146367467 ).
aresta( 15849 , 15847 , 0.31368580359198933 ).
aresta( 15849 , 15848 , 0.13551063745642336 ).
aresta( 15849 , 15850 , 0.3860535017505599 ).
aresta( 15849 , 15852 , 0.2459247692979649 ).
aresta( 15849 , 15853 , 0.15169421212392586 ).
aresta( 15849 , 21969 , 0.49489798666206536 ).
aresta( 15849 , 21932 , 0.147958693277262 ).
aresta( 15850 , 15849 , 0.3860535017505599 ).
aresta( 15850 , 15851 , 0.36018437553994054 ).
aresta( 15851 , 15845 , 0.5837577247647936 ).
aresta( 15851 , 15846 , 0.4667368555664502 ).
aresta( 15851 , 15847 , 0.35615194144645196 ).
aresta( 15851 , 15848 , 0.17417827864984836 ).
aresta( 15851 , 15850 , 0.36018437553994054 ).
aresta( 15851 , 15852 , 0.19993639263106336 ).
aresta( 15851 , 15853 , 0.09536546456835997 ).
aresta( 15851 , 21969 , 0.5382758060950951 ).
aresta( 15851 , 21932 , 0.18817312440769676 ).
aresta( 15852 , 15842 , 0.14522543385036255 ).
aresta( 15852 , 15843 , 0.10298461127008156 ).
aresta( 15852 , 15844 , 0.15205270842591265 ).
aresta( 15852 , 15849 , 0.2459247692979649 ).
aresta( 15852 , 15851 , 0.19993639263106336 ).
aresta( 15852 , 15855 , 0.15959842950071804 ).
aresta( 15852 , 15856 , 0.09938394564424093 ).
aresta( 15852 , 15857 , 0.0941361008971036 ).
aresta( 15852 , 15858 , 0.1576421812496329 ).
aresta( 15852 , 15859 , 0.07022138120806608 ).
aresta( 15852 , 15860 , 0.1161766452310488 ).
aresta( 15852 , 15861 , 0.17103437774793187 ).
aresta( 15852 , 15862 , 0.20147741832839866 ).
```

```
aresta( 15852 , 15863 , 0.23141536103519264 ).
aresta( 15852 , 15864 , 0.2596833846757338 ).
aresta( 15852 , 15868 , 0.108537427978599 ).
aresta( 15852 , 15869 , 0.054268543880543196 ).
aresta( 15853 , 15842 , 0.0403951048377961 ).
aresta( 15853 , 15843 , 0.04294517101494889 ).
aresta( 15853 , 15844 , 0.06786961771425697 ).
aresta( 15853 , 15849 , 0.15169421212392586 ).
aresta( 15853 , 15851 , 0.09536546456835997 ).
aresta( 15853 , 15855 , 0.25946579740232323 ).
aresta( 15853 , 15856 , 0.18890945964754557 ).
aresta( 15853 , 15857 , 0.18479541763553264 ).
aresta( 15853 , 15858 , 0.25838357439913107 ).
aresta( 15853 , 15859 , 0.126960257161009 ).
aresta( 15853 , 15860 , 0.07398874458389867 ).
aresta( 15853 , 15861 , 0.08892300878913083 ).
aresta( 15853 , 15862 , 0.11014496326963416 ).
aresta( 15853 , 15863 , 0.13406148693661643 ).
aresta( 15853 , 15864 , 0.1578717799247373 ).
aresta( 15853 , 15868 , 0.21490331442358357 ).
aresta( 15853 , 15869 , 0.16074539930146958 ).
aresta( 15855 , 15819 , 0.4720326243013291 ).
aresta( 15855 , 15820 , 0.5753687571226797 ).
aresta( 15855 , 15821 , 0.6360718804661127 ).
aresta( 15855 , 15822 , 0.6783060752581498 ).
aresta( 15855 , 15823 , 0.753862848718233 ).
aresta( 15855 , 15824 , 0.8198653929643684 ).
aresta( 15855 , 15841 , 0.39483878522466226 ).
aresta( 15855 , 15842 , 0.29293394757378866 ).
aresta( 15855 , 15843 , 0.2390850818696364 ).
aresta( 15855 , 15844 , 0.283135058264965 ).
aresta( 15855 , 15852 , 0.15959842950071804 ).
aresta( 15855 , 15853 , 0.25946579740232323 ).
aresta( 15855 , 15865 , 0.08707771635154579 ).
aresta( 15855 , 15866 , 0.04526751155965788 ).
aresta( 15855 , 15867 , 0.05180794516977309 ).
```

```
aresta( 15855 , 15871 , 0.13291531963501568 ).
aresta( 15855 , 15873 , 0.0825959259852509 ).
aresta( 15855 , 15875 , 0.05600368625314745 ).
aresta( 15855 , 15880 , 0.3108576526403782 ).
aresta( 15855 , 15885 , 0.1795399218128449 ).
aresta( 15855 , 15886 , 0.1530956517443407 ).
aresta( 15855 , 15887 , 0.10764160309577413 ).
aresta( 15855 , 21911 , 0.2848973762432177 ).
aresta( 15855 , 21952 , 0.632084886420952 ).
aresta( 15855 , 21889 , 0.29983228356513897 ).
aresta( 15856 , 15819 , 0.39706271012411476 ).
aresta( 15856 , 15820 , 0.5004734639164371 ).
aresta( 15856 , 15821 , 0.5612135923669008 ).
aresta( 15856 , 15822 , 0.6034896802091864 ).
aresta( 15856 , 15823 , 0.6791816986900128 ).
aresta( 15856 , 15824 , 0.7453487376782704 ).
aresta( 15856 , 15841 , 0.32107939130181684 ).
aresta( 15856 , 15842 , 0.22009994072900727 ).
aresta( 15856 , 15843 , 0.16507509265656722 ).
aresta( 15856 , 15844 , 0.2081563147881344 ).
aresta( 15856 , 15852 , 0.09938394564424093 ).
aresta( 15856 , 15853 , 0.18890945964754557 ).
aresta( 15856 , 15865 , 0.03778413340485162 ).
aresta( 15856 , 15866 , 0.06511328035025232 ).
aresta( 15856 , 15867 , 0.1228807587761164 ).
aresta( 15856 , 15871 , 0.11388962347238388 ).
aresta( 15856 , 15873 , 0.06142185312280037 ).
aresta( 15856 , 15875 , 0.0308524032410924 ).
aresta( 15856 , 15880 , 0.2357845582112767 ).
aresta( 15856 , 15885 , 0.10479076009908658 ).
aresta( 15856 , 15886 , 0.08216638333855561 ).
aresta( 15856 , 15887 , 0.037169526226228024 ).
aresta( 15856 , 21911 , 0.21102984792623408 ).
aresta( 15856 , 21952 , 0.557170279080958 ).
aresta( 15856 , 21889 , 0.2266870198774 ).
aresta( 15857 , 15819 , 0.39434488077375573 ).
```

```
aresta( 15857 , 15820 , 0.4976941652678774 ).
aresta( 15857 , 15821 , 0.5584075935459715 ).
aresta( 15857 , 15822 , 0.6006552163514353 ).
aresta( 15857 , 15823 , 0.6762655279634507 ).
aresta( 15857 , 15824 , 0.7423482502860496 ).
aresta( 15857 , 15841 , 0.31775278204456264 ).
aresta( 15857 , 15842 , 0.2164727101521401 ).
aresta( 15857 , 15843 , 0.16179091345760543 ).
aresta( 15857 , 15844 , 0.20544592305846868 ).
aresta( 15857 , 15852 , 0.0941361008971036 ).
aresta( 15857 , 15853 , 0.18479541763553264 ).
aresta( 15857 , 15865 , 0.03269622181126523 ).
aresta( 15857 , 15866 , 0.06425203746087774 ).
aresta( 15857 , 15867 , 0.12425286430121883 ).
aresta( 15857 , 15871 , 0.10910482651466481 ).
aresta( 15857 , 15873 , 0.05744493753977819 ).
aresta( 15857 , 15875 , 0.029784042259650062 ).
aresta( 15857 , 15880 , 0.23318243406242076 ).
aresta( 15857 , 15885 , 0.101873792561144 ).
aresta( 15857 , 15886 , 0.07791446604076367 ).
aresta( 15857 , 15887 , 0.03249393640830304 ).
aresta( 15857 , 21911 , 0.20771173411815602 ).
aresta( 15857 , 21952 , 0.5544067298142862 ).
aresta( 15857 , 21889 , 0.22314620335839874 ).
aresta( 15858 , 15819 , 0.472091176767095 ).
aresta( 15858 , 15820 , 0.575358485993852 ).
aresta( 15858 , 15821 , 0.6360303535951343 ).
aresta( 15858 , 15822 , 0.6782334621842718 ).
aresta( 15858 , 15823 , 0.7537040712583678 ).
aresta( 15858 , 15824 , 0.8196171241097907 ).
aresta( 15858 , 15841 , 0.39436075725389214 ).
aresta( 15858 , 15842 , 0.29225928238893584 ).
aresta( 15858 , 15843 , 0.23877540577370038 ).
aresta( 15858 , 15844 , 0.28324195794353657 ).
aresta( 15858 , 15852 , 0.1576421812496329 ).
aresta( 15858 , 15853 , 0.25838357439913107 ).
```

```
aresta( 15858 , 15865 , 0.0853572185626788 ).
aresta( 15858 , 15866 , 0.041184787124322365 ).
aresta( 15858 , 15867 , 0.049130358344356 ).
aresta( 15858 , 15871 , 0.12879444409057061 ).
aresta( 15858 , 15873 , 0.07932706126833107 ).
aresta( 15858 , 15875 , 0.05471701139979119 ).
aresta( 15858 , 15880 , 0.31104213423925064 ).
aresta( 15858 , 15885 , 0.17957785229706136 ).
aresta( 15858 , 15886 , 0.15232288141555253 ).
aresta( 15858 , 15887 , 0.10711834654895817 ).
aresta( 15858 , 21911 , 0.2845062776577587 ).
aresta( 15858 , 21952 , 0.632084070175833 ).
aresta( 15858 , 21889 , 0.29922851825170915 ).
aresta( 15859 , 15819 , 0.3285175000764331 ).
aresta( 15859 , 15820 , 0.43195476419185413 ).
aresta( 15859 , 15821 , 0.4927092891263625 ).
aresta( 15859 , 15822 , 0.5350040651998803 ).
aresta( 15859 , 15823 , 0.610766119711693 ).
aresta( 15859 , 15824 , 0.6770294530105255 ).
aresta( 15859 , 15841 , 0.25359293330711147 ).
aresta( 15859 , 15842 , 0.15416361382075708 ).
aresta( 15859 , 15843 , 0.09789797340501079 ).
aresta( 15859 , 15844 , 0.13962688444722474 ).
aresta( 15859 , 15852 , 0.07022138120806608 ).
aresta( 15859 , 15853 , 0.126960257161009 ).
aresta( 15859 , 15865 , 0.0732774466403984 ).
aresta( 15859 , 15866 , 0.12557447652402942 ).
aresta( 15859 , 15867 , 0.1898544249313748 ).
aresta( 15859 , 15871 , 0.1327774253153874 ).
aresta( 15859 , 15873 , 0.10183675030955 16 ).
aresta( 15859 , 15875 , 0.09298586558822036 ).
aresta( 15859 , 15880 , 0.16721665412005482 ).
aresta( 15859 , 15885 , 0.036889466177814585 ).
aresta( 15859 , 15886 , 0.03331452846188764 ).
aresta( 15859 , 15887 , 0.04201442608642302 ).
aresta( 15859 , 21911 , 0.14374499834841933 ).
```

```
aresta( 15859 , 21952 , 0.4886408108383201 ).
aresta( 15859 , 21889 , 0.16027331333081385 ).
aresta( 15860 , 15819 , 0.23897808732370565 ).
aresta( 15860 , 15820 , 0.34250874119523045 ).
aresta( 15860 , 15821 , 0.4033040859191861 ).
aresta( 15860 , 15822 , 0.4456599835002005 ).
aresta( 15860 , 15823 , 0.5216404706896802 ).
aresta( 15860 , 15824 , 0.5881672156384835 ).
aresta( 15860 , 15841 , 0.1682461769864437 ).
aresta( 15860 , 15842 , 0.07906637567176235 ).
aresta( 15860 , 15843 , 0.031335564883998135 ).
aresta( 15860 , 15844 , 0.05090574486761903 ).
aresta( 15860 , 15852 , 0.1161766452310488 ).
aresta( 15860 , 15853 , 0.07398874458389867 ).
aresta( 15860 , 15865 , 0.1588123349267948 ).
aresta( 15860 , 15866 , 0.21357985308215433 ).
aresta( 15860 , 15867 , 0.2795192082911087 ).
aresta( 15860 , 15871 , 0.19816614601338367 ).
aresta( 15860 , 15873 , 0.18426357167700083 ).
aresta( 15860 , 15875 , 0.182329569017323 ).
aresta( 15860 , 15880 , 0.07760971263121423 ).
aresta( 15860 , 15885 , 0.05583598118091623 ).
aresta( 15860 , 15886 , 0.09136515571328277 ).
aresta( 15860 , 15887 , 0.12945137236216817 ).
aresta( 15860 , 21911 , 0.06337851882447276 ).
aresta( 15860 , 21952 , 0.39912211161529376 ).
aresta( 15860 , 21889 , 0.082453526345236 ).
aresta( 15861 , 15819 , 0.17747567033198866 ).
aresta( 15861 , 15820 , 0.28097235879952154 ).
aresta( 15861 , 15821 , 0.3417625030310438 ).
aresta( 15861 , 15822 , 0.38416111051560636 ).
aresta( 15861 , 15823 , 0.46034256949224667 ).
aresta( 15861 , 15824 , 0.5271137275313654 ).
aresta( 15861 , 15841 , 0.11451511181979397 ).
aresta( 15861 , 15842 , 0.06216402071524202 ).
aresta( 15861 , 15843 , 0.0683397692679218 ).
```

```
aresta( 15861 , 15844 , 0.0212590249747395 ).
aresta( 15861 , 15852 , 0.17103437774793187 ).
aresta( 15861 , 15853 , 0.08892300878913083 ).
aresta( 15861 , 15865 , 0.22060331098819475 ).
aresta( 15861 , 15866 , 0.2756855629307303 ).
aresta( 15861 , 15867 , 0.3419919044564218 ).
aresta( 15861 , 15871 , 0.2534815159621031 ).
aresta( 15861 , 15873 , 0.24509116831876257 ).
aresta( 15861 , 15875 , 0.2447751827211076 ).
aresta( 15861 , 15880 , 0.019441939824023556 ).
aresta( 15861 , 15885 , 0.11796399772330643 ).
aresta( 15861 , 15886 , 0.1517758641057976 ).
aresta( 15861 , 15887 , 0.19174748609007636 ).
aresta( 15861 , 21911 , 0.046532193281265476 ).
aresta( 15861 , 21952 , 0.33744167313564827 ).
aresta( 15861 , 21889 , 0.058715201360529344 ).
aresta( 15862 , 15819 , 0.14462833972920425 ).
aresta( 15862 , 15820 , 0.24800961034295688 ).
aresta( 15862 , 15821 , 0.30877172596043106 ).
aresta( 15862 , 15822 , 0.35117878090543064 ).
aresta( 15862 , 15823 , 0.42743928437367085 ).
aresta( 15862 , 15824 , 0.4943144864730252 ).
aresta( 15862 , 15841 , 0.0892251278790514 ).
aresta( 15862 , 15842 , 0.0746049354829072 ).
aresta( 15862 , 15843 , 0.09856350339809834 ).
aresta( 15862 , 15844 , 0.049484839846525304 ).
aresta( 15862 , 15852 , 0.20147741832839866 ).
aresta( 15862 , 15853 , 0.11014496326963416 ).
aresta( 15862 , 15865 , 0.25345781589854455 ).
aresta( 15862 , 15866 , 0.30866856509435125 ).
aresta( 15862 , 15867 , 0.3752118136686142 ).
aresta( 15862 , 15871 , 0.2835013783052975 ).
aresta( 15862 , 15873 , 0.27747646051457703 ).
aresta( 15862 , 15875 , 0.27798096291894553 ).
aresta( 15862 , 15880 , 0.022715615537764317 ).
aresta( 15862 , 15885 , 0.15108378147090884 ).
```

```
aresta( 15862 , 15886 , 0.18423295703798934 ).
aresta( 15862 , 15887 , 0.22487272015010815 ).
aresta( 15862 , 21911 , 0.06492400994644329 ).
aresta( 15862 , 21952 , 0.3043834493694778 ).
aresta( 15862 , 21889 , 0.0681498833496397 ).
aresta( 15863 , 15819 , 0.11111017903365746 ).
aresta( 15863 , 15820 , 0.21441791357772708 ).
aresta( 15863 , 15821 , 0.27516928146405945 ).
aresta( 15863 , 15822 , 0.317577112305976 ).
aresta( 15863 , 15823 , 0.3938576199048924 ).
aresta( 15863 , 15824 , 0.46076757801448864 ).
aresta( 15863 , 15841 , 0.06711528483508904 ).
aresta( 15863 , 15842 , 0.09486691304906979 ).
aresta( 15863 , 15843 , 0.12929589401358457 ).
aresta( 15863 , 15844 , 0.08093960779329344 ).
aresta( 15863 , 15852 , 0.23141536103519264 ).
aresta( 15863 , 15853 , 0.13406148693661643 ).
aresta( 15863 , 15865 , 0.2858859866274978 ).
aresta( 15863 , 15866 , 0.34124242808375244 ).
aresta( 15863 , 15867 , 0.40817492370063424 ).
aresta( 15863 , 15871 , 0.31268966196218767 ).
aresta( 15863 , 15873 , 0.3092764537627247 ).
aresta( 15863 , 15875 , 0.31094328171927355 ).
aresta( 15863 , 15880 , 0.05282138797978117 ).
aresta( 15863 , 15885 , 0.18399907075726785 ).
aresta( 15863 , 15886 , 0.21628096792299478 ).
aresta( 15863 , 15887 , 0.2577215179933818 ).
aresta( 15863 , 21911 , 0.09022299491362913 ).
aresta( 15863 , 21952 , 0.2707683414989745 ).
aresta( 15863 , 21889 , 0.08716023329508608 ).
aresta( 15864 , 15819 , 0.07711857375903375 ).
aresta( 15864 , 15820 , 0.18062318201018374 ).
aresta( 15864 , 15821 , 0.24141913959830968 ).
aresta( 15864 , 15822 , 0.28381266595762344 ).
aresta( 15864 , 15823 , 0.3599854485118193 ).
aresta( 15864 , 15824 , 0.42678661205119306 ).
```

```
aresta( 15864 , 15841 , 0.05048065729409898 ).
aresta( 15864 , 15842 , 0.11750903479629167 ).
aresta( 15864 , 15843 , 0.15932215692885204 ).
aresta( 15864 , 15844 , 0.11246369988585665 ).
aresta( 15864 , 15852 , 0.2596833846757338 ).
aresta( 15864 , 15853 , 0.1578717799247373 ).
aresta( 15864 , 15865 , 0.31705563688375893 ).
aresta( 15864 , 15866 , 0.3725427154816892 ).
aresta( 15864 , 15867 , 0.4400534174092199 ).
aresta( 15864 , 15871 , 0.3397069266441516 ).
aresta( 15864 , 15873 , 0.33952237694359594 ).
aresta( 15864 , 15875 , 0.34290880179839955 ).
aresta( 15864 , 15880 , 0.0846191646764561 ).
aresta( 15864 , 15885 , 0.21613091260176423 ).
aresta( 15864 , 15886 , 0.24712149620077778 ).
aresta( 15864 , 15887 , 0.28957800854503624 ).
aresta( 15864 , 21911 , 0.11692194003216588 ).
aresta( 15864 , 21952 , 0.23714124978743958 ).
aresta( 15864 , 21889 , 0.10943982886610391 ).
aresta( 15865 , 15805 , 0.1301916442954011 ).
aresta( 15865 , 15806 , 0.1238785651929606 ).
aresta( 15865 , 15807 , 0.09014670342548893 ).
aresta( 15865 , 15855 , 0.08707771635154579 ).
aresta( 15865 , 15856 , 0.03778413340485162 ).
aresta( 15865 , 15857 , 0.03269622181126523 ).
aresta( 15865 , 15858 , 0.0853572185626788 ).
aresta( 15865 , 15859 , 0.0732774466403984 ).
aresta( 15865 , 15860 , 0.1588123349267948 ).
aresta( 15865 , 15861 , 0.22060331098819475 ).
aresta( 15865 , 15862 , 0.25345781589854455 ).
aresta( 15865 , 15863 , 0.2858859866274978 ).
aresta( 15865 , 15864 , 0.31705563688375893 ).
aresta( 15865 , 15868 , 0.06490640652462383 ).
aresta( 15865 , 15869 , 0.04242898790622027 ).
aresta( 15865 , 15871 , 0.076691839503633 ).
aresta( 15865 , 15873 , 0.029664744414314093 ).
```

```
aresta( 15865 , 15875 , 0.031481938100789 18 ).
aresta( 15865 , 15885 , 0.102988858151503 27 ).
aresta( 15865 , 15886 , 0.070044664873015 21 ).
aresta( 15865 , 15887 , 0.031366181826867 09 ).
aresta( 15865 , 21961 , 0.115440427825092 92 ).
aresta( 15866 , 15805 , 0.106456826620663 98 ).
aresta( 15866 , 15806 , 0.10259618749055 06 ).
aresta( 15866 , 15807 , 0.037173644504773916 ).
aresta( 15866 , 15855 , 0.04526751155965788 ).
aresta( 15866 , 15856 , 0.06511328035025232 ).
aresta( 15866 , 15857 , 0.06425203746087774 ).
aresta( 15866 , 15858 , 0.041184787124322365 ).
aresta( 15866 , 15859 , 0.12557447652402942 ).
aresta( 15866 , 15860 , 0.21357985308215433 ).
aresta( 15866 , 15861 , 0.2756855629307303 ).
aresta( 15866 , 15862 , 0.30866856509435125 ).
aresta( 15866 , 15863 , 0.34124242808375244 ).
aresta( 15866 , 15864 , 0.3725427154816892 ).
aresta( 15866 , 15868 , 0.04627415812356758 ).
aresta( 15866 , 15869 , 0.07576681631075724 ).
aresta( 15866 , 15871 , 0.08765170537604608 ).
aresta( 15866 , 15873 , 0.040157673151832655 ).
aresta( 15866 , 15875 , 0.034470350685720616 ).
aresta( 15866 , 15885 , 0.15778329157751103 ).
aresta( 15866 , 15886 , 0.1255574660859865 ).
aresta( 15866 , 15887 , 0.08425393365052601 ).
aresta( 15866 , 21961 , 0.07835533261131168 ).
aresta( 15867 , 15805 , 0.09972957970367313 ).
aresta( 15867 , 15806 , 0.10093126885579122 ).
aresta( 15867 , 15807 , 0.034746620782516384 ).
aresta( 15867 , 15855 , 0.05180794516977309 ).
aresta( 15867 , 15856 , 0.1228807587761164 ).
aresta( 15867 , 15857 , 0.12425286430121883 ).
aresta( 15867 , 15858 , 0.049130358344356 ).
aresta( 15867 , 15859 , 0.1898544249313748 ).
aresta( 15867 , 15860 , 0.2795192082911087 ).
```

```
aresta( 15867 , 15861 , 0.3419919044564218 ).
aresta( 15867 , 15862 , 0.3752118136686142 ).
aresta( 15867 , 15863 , 0.40817492370063424 ).
aresta( 15867 , 15864 , 0.4400534174092199 ).
aresta( 15867 , 15868 , 0.09759174954898811 ).
aresta( 15867 , 15869 , 0.14262435439609686 ).
aresta( 15867 , 15871 , 0.143849281147601 ).
aresta( 15867 , 15873 , 0.10807806089275473 ).
aresta( 15867 , 15875 , 0.09723587049806752 ).
aresta( 15867 , 15885 , 0.22417871620763533 ).
aresta( 15867 , 15886 , 0.19386766001398467 ).
aresta( 15867 , 15887 , 0.15049210654781625 ).
aresta( 15867 , 21961 , 0.06227702529701396 ).
aresta( 15868 , 15852 , 0.108537427978599 ).
aresta( 15868 , 15853 , 0.21490331442358357 ).
aresta( 15868 , 15865 , 0.06490640652462383 ).
aresta( 15868 , 15866 , 0.04627415812356758 ).
aresta( 15868 , 15867 , 0.09759174954898811 ).
aresta( 15868 , 15871 , 0.04630013386703766 ).
aresta( 15868 , 15873 , 0.035241846638323235 ).
aresta( 15868 , 15875 , 0.06637008107766351 ).
aresta( 15868 , 15883 , 0.08240483061843538 ).
aresta( 15868 , 15888 , 0.08868250417504311 ).
aresta( 15868 , 15889 , 0.07128624679594335 ).
aresta( 15868 , 15890 , 0.049209134148801015 ).
aresta( 15868 , 15891 , 0.04546574482963547 ).
aresta( 15869 , 15852 , 0.054268543880543196 ).
aresta( 15869 , 15853 , 0.16074539930146958 ).
aresta( 15869 , 15865 , 0.04242898790622027 ).
aresta( 15869 , 15866 , 0.07576681631075724 ).
aresta( 15869 , 15867 , 0.14262435439609686 ).
aresta( 15869 , 15871 , 0.03916688786056589 ).
aresta( 15869 , 15873 , 0.0356737391928049 ).
aresta( 15869 , 15875 , 0.06838332843615748 ).
aresta( 15869 , 15883 , 0.03258885240384509 ).
aresta( 15869 , 15888 , 0.14294078068907595 ).
```

```
aresta( 15869 , 15889 , 0.12522703797516213 ).
aresta( 15869 , 15890 , 0.1034547630010838 ).
aresta( 15869 , 15891 , 0.09720064093957348 ).
aresta( 15871 , 15855 , 0.13291531963501568 ).
aresta( 15871 , 15856 , 0.11388962347238388 ).
aresta( 15871 , 15857 , 0.10910482651466481 ).
aresta( 15871 , 15858 , 0.12879444409057061 ).
aresta( 15871 , 15859 , 0.1327774253153874 ).
aresta( 15871 , 15860 , 0.19816614601338367 ).
aresta( 15871 , 15861 , 0.2534815159621031 ).
aresta( 15871 , 15862 , 0.2835013783052975 ).
aresta( 15871 , 15863 , 0.31268966196218767 ).
aresta( 15871 , 15864 , 0.3397069266441516 ).
aresta( 15871 , 15865 , 0.076691839503633 ).
aresta( 15871 , 15866 , 0.08765170537604608 ).
aresta( 15871 , 15867 , 0.143849281147601 ).
aresta( 15871 , 15868 , 0.04630013386703766 ).
aresta( 15871 , 15869 , 0.03916688786056589 ).
aresta( 15871 , 15876 , 0.10768882154221518 ).
aresta( 15871 , 15877 , 0.165041681765399 ).
aresta( 15871 , 15878 , 0.18047531989987708 ).
aresta( 15871 , 15879 , 0.24257099008778604 ).
aresta( 15871 , 15898 , 0.04079301970903559 ).
aresta( 15871 , 15899 , 0.04818494465991521 ).
aresta( 15871 , 19301 , 0.03130742154942541 ).
aresta( 15873 , 15855 , 0.0825959259852509 ).
aresta( 15873 , 15856 , 0.06142185312280037 ).
aresta( 15873 , 15857 , 0.05744493753977819 ).
aresta( 15873 , 15858 , 0.07932706126833107 ).
aresta( 15873 , 15859 , 0.1018367503095516 ).
aresta( 15873 , 15860 , 0.18426357167700083 ).
aresta( 15873 , 15861 , 0.24509116831876257 ).
aresta( 15873 , 15862 , 0.27747646051457703 ).
aresta( 15873 , 15863 , 0.3092764537627247 ).
aresta( 15873 , 15864 , 0.33952237694359594 ).
aresta( 15873 , 15865 , 0.029664744414314093 ).
```

```
aresta( 15873 , 15866 , 0.040157673151832655 ).
aresta( 15873 , 15867 , 0.10807806089275473 ).
aresta( 15873 , 15868 , 0.035241846638323235 ).
aresta( 15873 , 15869 , 0.0356737391928049 ).
aresta( 15873 , 15876 , 0.14901454912377793 ).
aresta( 15873 , 15877 , 0.18222029314485974 ).
aresta( 15873 , 15878 , 0.17131612169077304 ).
aresta( 15873 , 15879 , 0.2249227860757105 ).
aresta( 15873 , 15898 , 0.09219212339569821 ).
aresta( 15873 , 15899 , 0.1028327347545128 ).
aresta( 15873 , 19301 , 0.02648127548402682 ).
aresta( 15875 , 15855 , 0.05600368625314745 ).
aresta( 15875 , 15856 , 0.0308524032410924 ).
aresta( 15875 , 15857 , 0.029784042259650062 ).
aresta( 15875 , 15858 , 0.05471701139979119 ).
aresta( 15875 , 15859 , 0.09298586558822036 ).
aresta( 15875 , 15860 , 0.182329569017323 ).
aresta( 15875 , 15861 , 0.2447751827211076 ).
aresta( 15875 , 15862 , 0.27798096291894553 ).
aresta( 15875 , 15863 , 0.31094328171927355 ).
aresta( 15875 , 15864 , 0.34290880179839955 ).
aresta( 15875 , 15865 , 0.03148193810078918 ).
aresta( 15875 , 15866 , 0.034470350685720616 ).
aresta( 15875 , 15867 , 0.09723587049806752 ).
aresta( 15875 , 15868 , 0.06637008107766351 ).
aresta( 15875 , 15869 , 0.06838332843615748 ).
aresta( 15875 , 15876 , 0.18323789944007599 ).
aresta( 15875 , 15877 , 0.20407508082947448 ).
aresta( 15875 , 15878 , 0.17730762330726624 ).
aresta( 15875 , 15879 , 0.22219290224288646 ).
aresta( 15875 , 15898 , 0.13045475124747824 ).
aresta( 15875 , 15899 , 0.14118816801447895 ).
aresta( 15875 , 19301 , 0.06492121777096818 ).
aresta( 15876 , 15809 , 0.19770728008010485 ).
aresta( 15876 , 15810 , 0.18421706529249796 ).
aresta( 15876 , 15871 , 0.10768882154221518 ).
```

```
aresta( 15876 , 15873 , 0.14901454912377793 ).
aresta( 15876 , 15875 , 0.18323789944007599 ).
aresta( 15876 , 15880 , 0.36012769057366517 ).
aresta( 15876 , 15888 , 0.11506655612682079 ).
aresta( 15876 , 15889 , 0.1215714100933009 ).
aresta( 15876 , 15890 , 0.11048137655747622 ).
aresta( 15876 , 15891 , 0.08840370745380327 ).
aresta( 15876 , 15892 , 0.2561338357003515 ).
aresta( 15876 , 15893 , 0.27655536776301143 ).
aresta( 15876 , 15894 , 0.2810744631213512 ).
aresta( 15876 , 15898 , 0.0694244735549071 ).
aresta( 15876 , 15899 , 0.10107235482233097 ).
aresta( 15877 , 15809 , 0.14963777685809218 ).
aresta( 15877 , 15810 , 0.11928150171583017 ).
aresta( 15877 , 15871 , 0.165041681765399 ).
aresta( 15877 , 15873 , 0.18222029314485974 ).
aresta( 15877 , 15875 , 0.20407508082947448 ).
aresta( 15877 , 15880 , 0.4261430660738482 ).
aresta( 15877 , 15888 , 0.08236115307972443 ).
aresta( 15877 , 15889 , 0.10242121032103496 ).
aresta( 15877 , 15890 , 0.10953341470150293 ).
aresta( 15877 , 15891 , 0.10213870577389741 ).
aresta( 15877 , 15892 , 0.18397288691615202 ).
aresta( 15877 , 15893 , 0.21212921791212627 ).
aresta( 15877 , 15894 , 0.2213601397414928 ).
aresta( 15877 , 15898 , 0.13964625920895768 ).
aresta( 15877 , 15899 , 0.17879675148271285 ).
aresta( 15878 , 15809 , 0.07251894481249586 ).
aresta( 15878 , 15810 , 0.036636644451277005 ).
aresta( 15878 , 15871 , 0.18047531989987708 ).
aresta( 15878 , 15873 , 0.17131612169077304 ).
aresta( 15878 , 15875 , 0.17730762330726624 ).
aresta( 15878 , 15880 , 0.4277704730890405 ).
aresta( 15878 , 15888 , 0.05550450149848368 ).
aresta( 15878 , 15889 , 0.07322435771082705 ).
aresta( 15878 , 15890 , 0.09450125050465376 ).
```

```
aresta( 15878 , 15891 , 0.10586329401752058 ).
aresta( 15878 , 15892 , 0.10516058102483505 ).
aresta( 15878 , 15893 , 0.13000149319044924 ).
aresta( 15878 , 15894 , 0.13871967995980078 ).
aresta( 15878 , 15898 , 0.1729502076290278 ).
aresta( 15878 , 15899 , 0.21235710034753166 ).
aresta( 15879 , 15809 , 0.08219095249619063 ).
aresta( 15879 , 15810 , 0.045785440164822215 ).
aresta( 15879 , 15871 , 0.24257099008778604 ).
aresta( 15879 , 15873 , 0.2249227860757105 ).
aresta( 15879 , 15875 , 0.22219290224288646 ).
aresta( 15879 , 15880 , 0.47942277576624326 ).
aresta( 15879 , 15888 , 0.11823812620783 93 ).
aresta( 15879 , 15889 , 0.13114186541318645 ).
aresta( 15879 , 15890 , 0.15395488000823127 ).
aresta( 15879 , 15891 , 0.16913599301459845 ).
aresta( 15879 , 15892 , 0.046944156613633756 ).
aresta( 15879 , 15893 , 0.08305257333218757 ).
aresta( 15879 , 15894 , 0.09854947964322712 ).
aresta( 15879 , 15898 , 0.23846030235483773 ).
aresta( 15879 , 15899 , 0.2773484080214388 ).
aresta( 15880 , 15855 , 0.3108576526403782 ).
aresta( 15880 , 15856 , 0.2357845582112767 ).
aresta( 15880 , 15857 , 0.23318243406242076 ).
aresta( 15880 , 15858 , 0.31104213423925064 ).
aresta( 15880 , 15859 , 0.16721665412005482 ).
aresta( 15880 , 15860 , 0.07760971263121423 ).
aresta( 15880 , 15861 , 0.019441939824023556 ).
aresta( 15880 , 15862 , 0.022715615537764317 ).
aresta( 15880 , 15863 , 0.05282138797978117 ).
aresta( 15880 , 15864 , 0.0846191646764561 ).
aresta( 15880 , 15876 , 0.36012769057366517 ).
aresta( 15880 , 15877 , 0.4261430660738482 ).
aresta( 15880 , 15878 , 0.4277704730890405 ).
aresta( 15880 , 15879 , 0.47942277576624326 ).
aresta( 15880 , 15881 , 0.3786516652806954 ).
```

```
aresta( 15880 , 19319 , 0.1586938208946766 ).
aresta( 15880 , 19415 , 0.3678598232224098 ).
aresta( 15881 , 15819 , 0.21728638852314078 ).
aresta( 15881 , 15820 , 0.1140099248192527 ).
aresta( 15881 , 15821 , 0.05387502584316203 ).
aresta( 15881 , 15822 , 0.016733552871422495 ).
aresta( 15881 , 15823 , 0.06931176597330761 ).
aresta( 15881 , 15824 , 0.13634615979374337 ).
aresta( 15881 , 15880 , 0.3786516652806954 ).
aresta( 15881 , 15882 , 0.11451870401884989 ).
aresta( 15881 , 21952 , 0.05729336588029037 ).
aresta( 15882 , 15819 , 0.33030776610379786 ).
aresta( 15882 , 15820 , 0.22678803874064105 ).
aresta( 15882 , 15821 , 0.1660076699219572 ).
aresta( 15882 , 15822 , 0.12360193206695314 ).
aresta( 15882 , 15823 , 0.0478983535936495 ).
aresta( 15882 , 15824 , 0.02561104196816848 ).
aresta( 15882 , 15881 , 0.11451870401884989 ).
aresta( 15882 , 15883 , 0.6867764532304594 ).
aresta( 15882 , 21952 , 0.17048817718405135 ).
aresta( 15883 , 15868 , 0.08240483061843538 ).
aresta( 15883 , 15869 , 0.03258885240384509 ).
aresta( 15883 , 15882 , 0.6867764532304594 ).
aresta( 15883 , 15884 , 0.049293934755860205 ).
aresta( 15883 , 15885 , 0.07993248540178052 ).
aresta( 15883 , 15886 , 0.04251062066254472 ).
aresta( 15883 , 15887 , 0.03565212416124829 ).
aresta( 15884 , 15883 , 0.049293934755860205 ).
aresta( 15884 , 15885 , 0.11342756472727161 ).
aresta( 15884 , 15886 , 0.07833485546060164 ).
aresta( 15884 , 15887 , 0.0847430499097465 ).
aresta( 15884 , 15898 , 0.07454566088397813 ).
aresta( 15884 , 15899 , 0.05353877046204756 ).
aresta( 15885 , 15842 , 0.11750447418894308 ).
aresta( 15885 , 15843 , 0.061029835492814634 ).
aresta( 15885 , 15844 , 0.10368105406846645 ).
```

```
aresta( 15885 , 15855 , 0.1795399218128449 ).
aresta( 15885 , 15856 , 0.10479076009908658 ).
aresta( 15885 , 15857 , 0.101873792561144 ).
aresta( 15885 , 15858 , 0.17957785229706136 ).
aresta( 15885 , 15859 , 0.036889466177814585 ).
aresta( 15885 , 15860 , 0.05583598118091623 ).
aresta( 15885 , 15861 , 0.11796399772330643 ).
aresta( 15885 , 15862 , 0.15108378147090884 ).
aresta( 15885 , 15863 , 0.18399907075726785 ).
aresta( 15885 , 15864 , 0.21613091260176423 ).
aresta( 15885 , 15865 , 0.10298885815150327 ).
aresta( 15885 , 15866 , 0.15778329157751103 ).
aresta( 15885 , 15867 , 0.22417871620763533 ).
aresta( 15885 , 15883 , 0.07993248540178052 ).
aresta( 15885 , 15884 , 0.11342756472727161 ).
aresta( 15885 , 15888 , 0.24700721652440075 ).
aresta( 15885 , 15889 , 0.22657083694142902 ).
aresta( 15885 , 15890 , 0.20767075508560684 ).
aresta( 15885 , 15891 , 0.20667703671017226 ).
aresta( 15885 , 15897 , 0.04521679503574678 ).
aresta( 15885 , 19319 , 0.03174009281352359 ).
aresta( 15885 , 19415 , 0.2365049003746667 ).
aresta( 15885 , 21911 , 0.10685554452058746 ).
aresta( 15885 , 21889 , 0.12348168325142685 ).
aresta( 15886 , 15842 , 0.13995754104762032 ).
aresta( 15886 , 15843 , 0.08797262490316851 ).
aresta( 15886 , 15844 , 0.1353424531786286 ).
aresta( 15886 , 15855 , 0.1530956517443407 ).
aresta( 15886 , 15856 , 0.08216638333855561 ).
aresta( 15886 , 15857 , 0.07791446604076367 ).
aresta( 15886 , 15858 , 0.15232288141555253 ).
aresta( 15886 , 15859 , 0.03331452846188764 ).
aresta( 15886 , 15860 , 0.09136515571328277 ).
aresta( 15886 , 15861 , 0.1517758641057976 ).
aresta( 15886 , 15862 , 0.18423295703798934 ).
aresta( 15886 , 15863 , 0.21628096792299478 ).
```

```
aresta( 15886 , 15864 , 0.24712149620077778 ).
aresta( 15886 , 15865 , 0.07004466487301521 ).
aresta( 15886 , 15866 , 0.1255574660859865 ).
aresta( 15886 , 15867 , 0.19386766001398467 ).
aresta( 15886 , 15883 , 0.04251062066254472 ).
aresta( 15886 , 15884 , 0.07833485546060164 ).
aresta( 15886 , 15888 , 0.2113889846555902 ).
aresta( 15886 , 15889 , 0.19148068458136688 ).
aresta( 15886 , 15890 , 0.17180171222831578 ).
aresta( 15886 , 15891 , 0.16966738783929186 ).
aresta( 15886 , 15897 , 0.05098775512570782 ).
aresta( 15886 , 19319 , 0.04054097300195912 ).
aresta( 15886 , 19415 , 0.20923674141466972 ).
aresta( 15886 , 21911 , 0.1329321326295728 ).
aresta( 15886 , 21889 , 0.1470052487524656 ).
aresta( 15887 , 15842 , 0.185311234178214 ).
aresta( 15887 , 15843 , 0.13173863034473365 ).
aresta( 15887 , 15844 , 0.1771427822486683 ).
aresta( 15887 , 15855 , 0.10764160309577413 ).
aresta( 15887 , 15856 , 0.037169526226228024 ).
aresta( 15887 , 15857 , 0.03249393640830304 ).
aresta( 15887 , 15858 , 0.10711834654895817 ).
aresta( 15887 , 15859 , 0.04201442608642302 ).
aresta( 15887 , 15860 , 0.12945137236216817 ).
aresta( 15887 , 15861 , 0.19174748609007636 ).
aresta( 15887 , 15862 , 0.22487272015010815 ).
aresta( 15887 , 15863 , 0.2577215179933818 ).
aresta( 15887 , 15864 , 0.28957800854503624 ).
aresta( 15887 , 15865 , 0.03136618182686709 ).
aresta( 15887 , 15866 , 0.08425393365052601 ).
aresta( 15887 , 15867 , 0.15049210654781625 ).
aresta( 15887 , 15883 , 0.03565212416124829 ).
aresta( 15887 , 15884 , 0.0847430499097465 ).
aresta( 15887 , 15888 , 0.17610395367579523 ).
aresta( 15887 , 15889 , 0.15495748433113227 ).
aresta( 15887 , 15890 , 0.1376807179086069 ).
```

```
aresta( 15887 , 15891 , 0.1405869589377386 ).
aresta( 15887 , 15897 , 0.09672507801449075 ).
aresta( 15887 , 19319 , 0.05353738567551384 ).
aresta( 15887 , 19415 , 0.16415797503105728 ).
aresta( 15887 , 21911 , 0.17739883871375056 ).
aresta( 15887 , 21889 , 0.19219072326047237 ).
aresta( 15888 , 15805 , 0.1673196539492436 ).
aresta( 15888 , 15806 , 0.1674777400224743 ).
aresta( 15888 , 15807 , 0.08446362231690586 ).
aresta( 15888 , 15811 , 0.10972759086694633 ).
aresta( 15888 , 15812 , 0.042890726135212186 ).
aresta( 15888 , 15868 , 0.08868250417504311 ).
aresta( 15888 , 15869 , 0.14294078068907595 ).
aresta( 15888 , 15876 , 0.11506655612682079 ).
aresta( 15888 , 15877 , 0.08236115307972443 ).
aresta( 15888 , 15878 , 0.05550450149848368 ).
aresta( 15888 , 15879 , 0.1182381262078393 ).
aresta( 15888 , 15885 , 0.24700721652440075 ).
aresta( 15888 , 15886 , 0.2113889846555902 ).
aresta( 15888 , 15887 , 0.17610395367579523 ).
aresta( 15888 , 15892 , 0.14881135010075242 ).
aresta( 15888 , 15893 , 0.16366600662701067 ).
aresta( 15888 , 15894 , 0.16665360322102843 ).
aresta( 15888 , 19301 , 0.11087014170898127 ).
aresta( 15888 , 21961 , 0.1297096410913056 ).
aresta( 15889 , 15805 , 0.14898335805463714 ).
aresta( 15889 , 15806 , 0.14856919004515645 ).
aresta( 15889 , 15807 , 0.06386484848295128 ).
aresta( 15889 , 15811 , 0.11412108558758242 ).
aresta( 15889 , 15812 , 0.041706195959626 ).
aresta( 15889 , 15868 , 0.07128624679594335 ).
aresta( 15889 , 15869 , 0.12522703797516213 ).
aresta( 15889 , 15876 , 0.1215714100933009 ).
aresta( 15889 , 15877 , 0.10242121032103496 ).
aresta( 15889 , 15878 , 0.07322435771082705 ).
aresta( 15889 , 15879 , 0.13114186541318645 ).
```

```
aresta( 15889 , 15885 , 0.22657083694142902 ).
aresta( 15889 , 15886 , 0.19148068458136688 ).
aresta( 15889 , 15887 , 0.15495748433113227 ).
aresta( 15889 , 15892 , 0.15638098268139627 ).
aresta( 15889 , 15893 , 0.16613779431619924 ).
aresta( 15889 , 15894 , 0.16655235754534622 ).
aresta( 15889 , 19301 , 0.0937332231302844 ).
aresta( 15889 , 21961 , 0.11168882731081746 ).
aresta( 15890 , 15805 , 0.15156149079737344 ).
aresta( 15890 , 15806 , 0.1499947129450299 ).
aresta( 15890 , 15807 , 0.06413044736620271 ).
aresta( 15890 , 15811 , 0.13619181012302536 ).
aresta( 15890 , 15812 , 0.06354301556764921 ).
aresta( 15890 , 15868 , 0.049209134148801015 ).
aresta( 15890 , 15869 , 0.1034547630010838 ).
aresta( 15890 , 15876 , 0.11048137655747622 ).
aresta( 15890 , 15877 , 0.10953341470150293 ).
aresta( 15890 , 15878 , 0.09450125050465376 ).
aresta( 15890 , 15879 , 0.15395488000823127 ).
aresta( 15890 , 15885 , 0.20767075508560684 ).
aresta( 15890 , 15886 , 0.17180171222831578 ).
aresta( 15890 , 15887 , 0.1376807179086069 ).
aresta( 15890 , 15892 , 0.17898180560252866 ).
aresta( 15890 , 15893 , 0.1873304546535548 ).
aresta( 15890 , 15894 , 0.1866589039472142 ).
aresta( 15890 , 19301 , 0.07157482291057174 ).
aresta( 15890 , 21961 , 0.1157220734687291 ).
aresta( 15891 , 15805 , 0.17214242441866054 ).
aresta( 15891 , 15806 , 0.1700288249003034 ).
aresta( 15891 , 15807 , 0.08518044310516185 ).
aresta( 15891 , 15811 , 0.15626047428911907 ).
aresta( 15891 , 15812 , 0.08408062018952182 ).
aresta( 15891 , 15868 , 0.04546574482963547 ).
aresta( 15891 , 15869 , 0.09720064093957348 ).
aresta( 15891 , 15876 , 0.08840370745380327 ).
aresta( 15891 , 15877 , 0.10213870577389741 ).
```

```
aresta( 15891 , 15878 , 0.10586329401752058 ).
aresta( 15891 , 15879 , 0.16913599301459845 ).
aresta( 15891 , 15885 , 0.20667703671017226 ).
aresta( 15891 , 15886 , 0.16966738783929186 ).
aresta( 15891 , 15887 , 0.1405869589377386 ).
aresta( 15891 , 15892 , 0.19773967198449427 ).
aresta( 15891 , 15893 , 0.20853069459843251 ).
aresta( 15891 , 15894 , 0.20875782677722596 ).
aresta( 15891 , 19301 , 0.06524598956516026 ).
aresta( 15891 , 21961 , 0.13724419399299004 ).
aresta( 15892 , 15808 , 0.07473525468785382 ).
aresta( 15892 , 15811 , 0.0452414012283684 ).
aresta( 15892 , 15812 , 0.11586472587586745 ).
aresta( 15892 , 15876 , 0.2561338357003515 ).
aresta( 15892 , 15877 , 0.18397288691615202 ).
aresta( 15892 , 15878 , 0.10516058102483505 ).
aresta( 15892 , 15879 , 0.046944156613633756 ).
aresta( 15892 , 15888 , 0.14881135010075242 ).
aresta( 15892 , 15889 , 0.15638098268139627 ).
aresta( 15892 , 15890 , 0.17898180560252866 ).
aresta( 15892 , 15891 , 0.19773967198449427 ).
aresta( 15892 , 15896 , 0.8993877959320543 ).
aresta( 15893 , 15808 , 0.059987607279522556 ).
aresta( 15893 , 15811 , 0.05448062104408998 ).
aresta( 15893 , 15812 , 0.12446235599839334 ).
aresta( 15893 , 15876 , 0.27655536776301143 ).
aresta( 15893 , 15877 , 0.21212921791212627 ).
aresta( 15893 , 15878 , 0.13000149319044924 ).
aresta( 15893 , 15879 , 0.08305257333218757 ).
aresta( 15893 , 15888 , 0.16366600662701067 ).
aresta( 15893 , 15889 , 0.16613779431619924 ).
aresta( 15893 , 15890 , 0.1873304546535548 ).
aresta( 15893 , 15891 , 0.20853069459843251 ).
aresta( 15893 , 15896 , 0.8894874521496071 ).
aresta( 15894 , 15808 , 0.05362319943152387 ).
aresta( 15894 , 15811 , 0.061560407978865146 ).
```

```
aresta( 15894 , 15812 , 0.12546982148339292 ).
aresta( 15894 , 15876 , 0.2810744631213512 ).
aresta( 15894 , 15877 , 0.2213601397414928 ).
aresta( 15894 , 15878 , 0.13871967995980078 ).
aresta( 15894 , 15879 , 0.09854947964322712 ).
aresta( 15894 , 15888 , 0.16665360322102843 ).
aresta( 15894 , 15889 , 0.16655235754534622 ).
aresta( 15894 , 15890 , 0.1866589039472142 ).
aresta( 15894 , 15891 , 0.20875782677722596 ).
aresta( 15894 , 15896 , 0.8781084713301271 ).
aresta( 15896 , 15819 , 0.25019095789108836 ).
aresta( 15896 , 15820 , 0.14674335695660054 ).
aresta( 15896 , 15821 , 0.08611844636089686 ).
aresta( 15896 , 15822 , 0.044509042298095124 ).
aresta( 15896 , 15823 , 0.03778678611000041 ).
aresta( 15896 , 15824 , 0.10371423546551421 ).
aresta( 15896 , 15892 , 0.8993877959320543 ).
aresta( 15896 , 15893 , 0.8894874521496071 ).
aresta( 15896 , 15894 , 0.8781084713301271 ).
aresta( 15896 , 15897 , 0.5241430990482939 ).
aresta( 15896 , 21952 , 0.09004337779012257 ).
aresta( 15897 , 15885 , 0.04521679503574678 ).
aresta( 15897 , 15886 , 0.05098775512570782 ).
aresta( 15897 , 15887 , 0.09672507801449075 ).
aresta( 15897 , 15896 , 0.5241430990482939 ).
aresta( 15897 , 15898 , 0.17153574917312225 ).
aresta( 15897 , 15899 , 0.14326390596924066 ).
aresta( 15898 , 15834 , 0.6749790391109016 ).
aresta( 15898 , 15836 , 0.7026036594983183 ).
aresta( 15898 , 15871 , 0.04079301970903559 ).
aresta( 15898 , 15873 , 0.09219212339569821 ).
aresta( 15898 , 15875 , 0.13045475124747824 ).
aresta( 15898 , 15876 , 0.0694244735549071 ).
aresta( 15898 , 15877 , 0.13964625920895768 ).
aresta( 15898 , 15878 , 0.1729502076290278 ).
aresta( 15898 , 15879 , 0.23846030235483773 ).
```

```
aresta( 15898 , 15884 , 0.07454566088397813 ).
aresta( 15898 , 15888 , 0.1210295472397542 ).
aresta( 15898 , 15889 , 0.11455477017423668 ).
aresta( 15898 , 15890 , 0.09373746432489961 ).
aresta( 15898 , 15891 , 0.07221139021923356 ).
aresta( 15898 , 15897 , 0.17153574917312225 ).
aresta( 15898 , 19365 , 0.970193812726979 ).
aresta( 15898 , 21866 , 0.6738183846135747 ).
aresta( 15899 , 15834 , 0.6350368491880308 ).
aresta( 15899 , 15836 , 0.6624122852372273 ).
aresta( 15899 , 15871 , 0.04818494465991521 ).
aresta( 15899 , 15873 , 0.1028327347545128 ).
aresta( 15899 , 15875 , 0.14118816801447895 ).
aresta( 15899 , 15876 , 0.10107235482233097 ).
aresta( 15899 , 15877 , 0.17879675148271285 ).
aresta( 15899 , 15878 , 0.21235710034753166 ).
aresta( 15899 , 15879 , 0.2773484080214388 ).
aresta( 15899 , 15884 , 0.05353877046204756 ).
aresta( 15899 , 15888 , 0.1592994553855701 ).
aresta( 15899 , 15889 , 0.1504718205052643 ).
aresta( 15899 , 15890 , 0.12838856225909914 ).
aresta( 15899 , 15891 , 0.1090005335752892 ).
aresta( 15899 , 15897 , 0.14326390596924066 ).
aresta( 15899 , 19365 , 0.9495833858454429 ).
aresta( 15899 , 21866 , 0.6488542472027303 ).
aresta( 19365 , 15898 , 0.970193812726979 ).
aresta( 19365 , 15899 , 0.9495833858454429 ).
aresta( 19365 , 19216 , 0.48962019476559066 ).
aresta( 19365 , 21961 , 0.8881919645173861 ).
aresta( 19365 , 21966 , 0.46252663325573845 ).
aresta( 19216 , 19365 , 0.48962019476559066 ).
aresta( 19216 , 19280 , 0.6656593985885294 ).
aresta( 19216 , 21866 , 0.6480111838006217 ).
aresta( 19280 , 19216 , 0.6656593985885294 ).
aresta( 19280 , 19301 , 0.5976639790672955 ).
aresta( 19301 , 15871 , 0.03130742154942541 ).
```

```
aresta( 19301 , 15873 , 0.026481275484026 82 ).
aresta( 19301 , 15875 , 0.064921217770968 18 ).
aresta( 19301 , 15888 , 0.110870141708981 27 ).
aresta( 19301 , 15889 , 0.093733223130284 4 ).
aresta( 19301 , 15890 , 0.071574822910571 74 ).
aresta( 19301 , 15891 , 0.065245989565160 26 ).
aresta( 19301 , 19280 , 0.59766397906729 55 ).
aresta( 19301 , 19315 , 0.35386810687176 28 ).
aresta( 19301 , 21854 , 0.263268207376668 06 ).
aresta( 19315 , 19301 , 0.35386810687176 28 ).
aresta( 19315 , 19300 , 0.097422940602982 43 ).
aresta( 19315 , 19371 , 0.08426545953396 85 ).
aresta( 19315 , 21969 , 0.83815013938224 54 ).
aresta( 19315 , 21938 , 0.167547428668049 72 ).
aresta( 19315 , 21932 , 0.52983558191519 36 ).
aresta( 19315 , 21939 , 0.176235679422205 72 ).
aresta( 19300 , 19315 , 0.097422940602982 43 ).
aresta( 19300 , 19407 , 0.190477869289198 7 ).
aresta( 19300 , 19295 , 0.37400130069760 12 ).
aresta( 19300 , 19390 , 0.84279238462853 23 ).
aresta( 19300 , 19337 , 0.196807012335438 24 ).
aresta( 19300 , 21854 , 0.179178620249143 86 ).
aresta( 19407 , 19300 , 0.190477869289198 7 ).
aresta( 19407 , 19282 , 0.83457157799935 54 ).
aresta( 19407 , 19257 , 1.10174992704854 22 ).
aresta( 19407 , 19319 , 0.195649408535875 67 ).
aresta( 19407 , 19236 , 1.20658869619551 06 ).
aresta( 19407 , 19371 , 0.17470483985987 68 ).
aresta( 19407 , 19415 , 0.25097605538615 46 ).
aresta( 19407 , 21843 , 0.97561914709545 43 ).
aresta( 19282 , 15819 , 0.413964201189275 57 ).
aresta( 19282 , 15820 , 0.311544081733033 05 ).
aresta( 19282 , 15821 , 0.25179682042254 2 ).
aresta( 19282 , 15822 , 0.210105874398892 25 ).
aresta( 19282 , 15823 , 0.134583412097459 94 ).
aresta( 19282 , 15824 , 0.068376285602433 93 ).
```

```
aresta( 19282 , 19407 , 0.8345715779993554 ).
aresta( 19282 , 19257 , 0.5334083932594308 ).
aresta( 19282 , 19236 , 0.6757141410343085 ).
aresta( 19282 , 19337 , 0.8366167520894118 ).
aresta( 19282 , 21843 , 0.23923897373096942 ).
aresta( 19257 , 19407 , 1.1017499270485422 ).
aresta( 19257 , 19282 , 0.5334083932594308 ).
aresta( 19257 , 19293 , 0.34156329288770465 ).
aresta( 19257 , 19319 , 1.0860335552000662 ).
aresta( 19257 , 19295 , 0.947219385071839 ).
aresta( 19257 , 19415 , 1.2905124613168144 ).
aresta( 19257 , 19337 , 1.1064214229125315 ).
aresta( 19257 , 21844 , 0.22975954703870904 ).
aresta( 19293 , 19257 , 0.34156329288770465 ).
aresta( 19293 , 19310 , 1.5171841759384006 ).
aresta( 19293 , 19236 , 0.2172360339286054 ).
aresta( 19293 , 21843 , 0.6471953046259731 ).
aresta( 19310 , 19293 , 1.5171841759384006 ).
aresta( 19310 , 19319 , 0.2823752830423265 ).
aresta( 19310 , 19415 , 0.10311311166276078 ).
aresta( 19319 , 15842 , 0.14882548825152334 ).
aresta( 19319 , 15843 , 0.09201970889419547 ).
aresta( 19319 , 15844 , 0.13145411432242124 ).
aresta( 19319 , 15880 , 0.1586938208946766 ).
aresta( 19319 , 15885 , 0.03174009281352359 ).
aresta( 19319 , 15886 , 0.04054097300195912 ).
aresta( 19319 , 15887 , 0.05353738567551384 ).
aresta( 19319 , 19407 , 0.19564940853587567 ).
aresta( 19319 , 19257 , 1.0860335552000662 ).
aresta( 19319 , 19310 , 0.2823752830423265 ).
aresta( 19319 , 19236 , 1.2056872087602124 ).
aresta( 19319 , 19278 , 1.0116404877600325 ).
aresta( 19319 , 19337 , 0.1920814251554384 ).
aresta( 19319 , 21843 , 0.9094781735763621 ).
aresta( 19236 , 19407 , 1.2065886961955106 ).
aresta( 19236 , 19282 , 0.6757141410343085 ).
```

```
aresta( 19236 , 19293 , 0.2172360339286054 ).
aresta( 19236 , 19319 , 1.2056872087602124 ).
aresta( 19236 , 19295 , 0.9961713952026731 ).
aresta( 19236 , 19415 , 1.4067295914189637 ).
aresta( 19236 , 19337 , 1.2116167056292633 ).
aresta( 19236 , 21844 , 0.3533454807155718 ).
aresta( 19295 , 19300 , 0.3740013006976012 ).
aresta( 19295 , 19257 , 0.947219385071839 ).
aresta( 19295 , 19236 , 0.9961713952026731 ).
aresta( 19295 , 19371 , 0.37333690521007695 ).
aresta( 19295 , 21843 , 0.9811884958903644 ).
aresta( 19371 , 19315 , 0.0842654595339685 ).
aresta( 19371 , 19407 , 0.1747048398598768 ).
aresta( 19371 , 19295 , 0.37333690521007695 ).
aresta( 19371 , 19390 , 0.8703241452819422 ).
aresta( 19371 , 19337 , 0.1811547697550659 ).
aresta( 19371 , 21854 , 0.17259799095328932 ).
aresta( 19390 , 19300 , 0.8427923846285323 ).
aresta( 19390 , 19371 , 0.8703241452819422 ).
aresta( 19390 , 19278 , 0.26133482572940414 ).
aresta( 19278 , 19319 , 1.0116404877600325 ).
aresta( 19278 , 19390 , 0.26133482572940414 ).
aresta( 19278 , 19415 , 1.2199728110034036 ).
aresta( 19415 , 15842 , 0.34919412869573424 ).
aresta( 19415 , 15843 , 0.29582623142849684 ).
aresta( 19415 , 15844 , 0.34012548312537816 ).
aresta( 19415 , 15880 , 0.3678598232224098 ).
aresta( 19415 , 15885 , 0.2365049003746667 ).
aresta( 19415 , 15886 , 0.20923674141466972 ).
aresta( 19415 , 15887 , 0.16415797503105728 ).
aresta( 19415 , 19407 , 0.2509760553861546 ).
aresta( 19415 , 19257 , 1.2905124613168144 ).
aresta( 19415 , 19310 , 0.10311311166276078 ).
aresta( 19415 , 19236 , 1.4067295914189637 ).
aresta( 19415 , 19278 , 1.2199728110034036 ).
aresta( 19415 , 19337 , 0.24450338577715253 ).
```

```
aresta( 19415 , 21843 , 1.119281873005917 ).
aresta( 19337 , 19300 , 0.19680701233543824 ).
aresta( 19337 , 19282 , 0.8366167520894118 ).
aresta( 19337 , 19257 , 1.1064214229125315 ).
aresta( 19337 , 19319 , 0.1920814251554384 ).
aresta( 19337 , 19236 , 1.2116167056292633 ).
aresta( 19337 , 19371 , 0.1811547697550659 ).
aresta( 19337 , 19415 , 0.24450338577715253 ).
aresta( 19337 , 21843 , 0.9789208390841768 ).
aresta( 21843 , 19407 , 0.9756191470954543 ).
aresta( 21843 , 19282 , 0.23923897373096942 ).
aresta( 21843 , 19293 , 0.6471953046259731 ).
aresta( 21843 , 19319 , 0.9094781735763621 ).
aresta( 21843 , 19295 , 0.9811884958903644 ).
aresta( 21843 , 19415 , 1.119281873005917 ).
aresta( 21843 , 19337 , 0.9789208390841768 ).
aresta( 21843 , 21844 , 0.2415132641647703 ).
aresta( 21844 , 19257 , 0.22975954703870904 ).
aresta( 21844 , 19236 , 0.3533454807155718 ).
aresta( 21844 , 21843 , 0.2415132641647703 ).
aresta( 21844 , 21957 , 0.8322377231199557 ).
aresta( 21957 , 21844 , 0.8322377231199557 ).
aresta( 21957 , 21959 , 0.47232203587217636 ).
aresta( 21959 , 15819 , 0.34584157941968247 ).
aresta( 21959 , 15820 , 0.24522429471152277 ).
aresta( 21959 , 15821 , 0.18757194958837162 ).
aresta( 21959 , 15822 , 0.1481206104511038 ).
aresta( 21959 , 15823 , 0.08063240937530029 ).
aresta( 21959 , 15824 , 0.04655616646221061 ).
aresta( 21959 , 15845 , 0.16439382208225722 ).
aresta( 21959 , 15846 , 0.09780519193824547 ).
aresta( 21959 , 15847 , 0.133046278514393 ).
aresta( 21959 , 15848 , 0.2937153532289583 ).
aresta( 21959 , 21957 , 0.47232203587217636 ).
aresta( 21959 , 21911 , 0.5258492526375472 ).
aresta( 21959 , 21889 , 0.5108953233336369 ).
```

```
aresta( 21911 , 15842 , 0.017953208396719114 ).
aresta( 21911 , 15843 , 0.04596337836530813 ).
aresta( 21911 , 15844 , 0.02775770717175955 ).
aresta( 21911 , 15855 , 0.2848973762432177 ).
aresta( 21911 , 15856 , 0.21102984792623408 ).
aresta( 21911 , 15857 , 0.20771173411815602 ).
aresta( 21911 , 15858 , 0.2845062776577587 ).
aresta( 21911 , 15859 , 0.14374499834841933 ).
aresta( 21911 , 15860 , 0.06337851882447276 ).
aresta( 21911 , 15861 , 0.046532193281265476 ).
aresta( 21911 , 15862 , 0.06492400994644329 ).
aresta( 21911 , 15863 , 0.09022299491362913 ).
aresta( 21911 , 15864 , 0.11692194003216588 ).
aresta( 21911 , 15885 , 0.10685554452058746 ).
aresta( 21911 , 15886 , 0.1329321326295728 ).
aresta( 21911 , 15887 , 0.17739883871375056 ).
aresta( 21911 , 21959 , 0.5258492526375472 ).
aresta( 21911 , 21952 , 0.34890244135217013 ).
aresta( 21911 , 21969 , 0.6117266677515839 ).
aresta( 21911 , 21932 , 0.26495826414119655 ).
aresta( 21911 , 21925 , 0.4010053452884641 ).
aresta( 21952 , 15818 , 0.02408648236961493 ).
aresta( 21952 , 15834 , 0.27379692159698965 ).
aresta( 21952 , 15836 , 0.19581610553138332 ).
aresta( 21952 , 15855 , 0.632084886420952 ).
aresta( 21952 , 15856 , 0.557170279080958 ).
aresta( 21952 , 15857 , 0.5544067298142862 ).
aresta( 21952 , 15858 , 0.632084070175833 ).
aresta( 21952 , 15859 , 0.4886408108383201 ).
aresta( 21952 , 15860 , 0.39912211161529376 ).
aresta( 21952 , 15861 , 0.33744167313564827 ).
aresta( 21952 , 15862 , 0.3043834493694778 ).
aresta( 21952 , 15863 , 0.2707683414989745 ).
aresta( 21952 , 15864 , 0.23714124978743958 ).
aresta( 21952 , 15881 , 0.05729336588029037 ).
aresta( 21952 , 15882 , 0.17048817718405135 ).
```

```
aresta( 21952 , 15896 , 0.09004337779012257 ).
aresta( 21952 , 21911 , 0.34890244135217013 ).
aresta( 21952 , 21969 , 0.3056249333937886 ).
aresta( 21952 , 21932 , 0.1636067304479345 ).
aresta( 21952 , 21889 , 0.33652294044647174 ).
aresta( 21969 , 15830 , 0.077361732777701 ).
aresta( 21969 , 15831 , 0.032767950353301496 ).
aresta( 21969 , 15834 , 0.15821868148902074 ).
aresta( 21969 , 15836 , 0.11585726005593713 ).
aresta( 21969 , 15849 , 0.49489798666206536 ).
aresta( 21969 , 15851 , 0.5382758060950951 ).
aresta( 21969 , 19315 , 0.8381501393822454 ).
aresta( 21969 , 21911 , 0.6117266677515839 ).
aresta( 21969 , 21952 , 0.3056249333937886 ).
aresta( 21969 , 21854 , 0.8096613664178691 ).
aresta( 21969 , 21944 , 0.9232158804113518 ).
aresta( 21969 , 21889 , 0.5950310256127412 ).
aresta( 21854 , 19301 , 0.26326820737666806 ).
aresta( 21854 , 19300 , 0.17917862024914386 ).
aresta( 21854 , 19371 , 0.17259799095328932 ).
aresta( 21854 , 21969 , 0.8096613664178691 ).
aresta( 21854 , 21938 , 0.2578552311419206 ).
aresta( 21854 , 21932 , 0.4825322823548149 ).
aresta( 21854 , 21939 , 0.26664385147878156 ).
aresta( 21938 , 19315 , 0.16754742866804972 ).
aresta( 21938 , 21854 , 0.2578552311419206 ).
aresta( 21938 , 21961 , 0.4387457033448275 ).
aresta( 21938 , 21927 , 0.24789329536539187 ).
aresta( 21938 , 21949 , 0.9402148356705787 ).
aresta( 21961 , 15865 , 0.11544042782509292 ).
aresta( 21961 , 15866 , 0.07835533261131168 ).
aresta( 21961 , 15867 , 0.06227702529701396 ).
aresta( 21961 , 15888 , 0.1297096410913056 ).
aresta( 21961 , 15889 , 0.11168882731081746 ).
aresta( 21961 , 15890 , 0.1157220734687291 ).
aresta( 21961 , 15891 , 0.13724419399299004 ).
```

```
aresta( 21961 , 19365 , 0.8881919645173861 ).
aresta( 21961 , 21938 , 0.4387457033448275 ).
aresta( 21961 , 21866 , 0.6255565855026801 ).
aresta( 21961 , 21939 , 0.4475047113897471 ).
aresta( 21866 , 15898 , 0.6738183846135747 ).
aresta( 21866 , 15899 , 0.6488542472027303 ).
aresta( 21866 , 19216 , 0.6480111838006217 ).
aresta( 21866 , 21961 , 0.6255565855026801 ).
aresta( 21866 , 21966 , 0.1637650861192788 ).
aresta( 21966 , 19365 , 0.46252663325573845 ).
aresta( 21966 , 21866 , 0.1637650861192788 ).
aresta( 21966 , 21967 , 0.16057617510499167 ).
aresta( 21967 , 21966 , 0.16057617510499167 ).
aresta( 21967 , 21944 , 0.3839708290301611 ).
aresta( 21944 , 15805 , 0.04817410037986708 ).
aresta( 21944 , 15806 , 0.04999336901070282 ).
aresta( 21944 , 15807 , 0.046755071106155566 ).
aresta( 21944 , 21969 , 0.9232158804113518 ).
aresta( 21944 , 21967 , 0.3839708290301611 ).
aresta( 21944 , 21932 , 0.5734188249458578 ).
aresta( 21932 , 15830 , 0.35930582938194205 ).
aresta( 21932 , 15831 , 0.3317091939197365 ).
aresta( 21932 , 15834 , 0.23991222282629562 ).
aresta( 21932 , 15836 , 0.23908570935167112 ).
aresta( 21932 , 15849 , 0.147958693277262 ).
aresta( 21932 , 15851 , 0.18817312440769676 ).
aresta( 21932 , 19315 , 0.5298355819151936 ).
aresta( 21932 , 21911 , 0.26495826414119655 ).
aresta( 21932 , 21952 , 0.1636067304479345 ).
aresta( 21932 , 21854 , 0.4825322823548149 ).
aresta( 21932 , 21944 , 0.5734188249458578 ).
aresta( 21932 , 21889 , 0.2472402693273157 ).
aresta( 21889 , 15842 , 0.008078567212858094 ).
aresta( 21889 , 15843 , 0.06254799163777187 ).
aresta( 21889 , 15844 , 0.043767652984213354 ).
aresta( 21889 , 15855 , 0.29983228356513897 ).
```

```
aresta( 21889 , 15856 , 0.2266870198774 ).
aresta( 21889 , 15857 , 0.22314620335839874 ).
aresta( 21889 , 15858 , 0.29922851825170915 ).
aresta( 21889 , 15859 , 0.16027331333081385 ).
aresta( 21889 , 15860 , 0.082453526345236 ).
aresta( 21889 , 15861 , 0.058715201360529344 ).
aresta( 21889 , 15862 , 0.0681498833496397 ).
aresta( 21889 , 15863 , 0.08716023329508608 ).
aresta( 21889 , 15864 , 0.10943982886610391 ).
aresta( 21889 , 15885 , 0.12348168325142685 ).
aresta( 21889 , 15886 , 0.1470052487524656 ).
aresta( 21889 , 15887 , 0.19219072326047237 ).
aresta( 21889 , 21959 , 0.5108953233336369 ).
aresta( 21889 , 21952 , 0.33652294044647174 ).
aresta( 21889 , 21969 , 0.5950310256127412 ).
aresta( 21889 , 21932 , 0.2472402693273157 ).
aresta( 21889 , 21925 , 0.41968555383450146 ).
aresta( 21925 , 21911 , 0.4010053452884641 ).
aresta( 21925 , 21889 , 0.41968555383450146 ).
aresta( 21925 , 21927 , 0.3839395481021777 ).
aresta( 21927 , 21938 , 0.24789329536539187 ).
aresta( 21927 , 21925 , 0.3839395481021777 ).
aresta( 21927 , 21939 , 0.25339779723108086 ).
aresta( 21939 , 19315 , 0.17623567942220572 ).
aresta( 21939 , 21854 , 0.26664385147878156 ).
aresta( 21939 , 21961 , 0.4475047113897471 ).
aresta( 21939 , 21927 , 0.25339779723108086 ).
aresta( 21939 , 21949 , 0.9376823647077369 ).
aresta( 21949 , 21938 , 0.9402148356705787 ).
aresta( 21949 , 21939 , 0.9376823647077369 ).
```