Unicamp – Universidade Estadual de Campinas
FT – Faculdade de Tecnologia
Limeira – São Paulo

Guia Ilustrativo para conversão de códigos IF1 para Grafos na Ferramenta Gephi

1 Identificação

Autor: **Renan Eugênio da Silva**

Revisão: **Prof. Dr. André F. de Angelis**

Instituição Sede: **Universidade Estadual de Campinas**

Local de Execução: **Faculdade de Tecnologia – Limeira/SP**

Área: **Sistemas de Informação e Comunicação**

Objetivo

Este guia tem como finalidade descrever os passos necessários para a tradução de arquivos intermediários gerados a partir de código Sisal compilados em grafos que possam ser exportados para uma ferramenta que faz a análise e processamento.

Como premissa para compreender o processo que é descrito neste documento, é necessário ter um entendimento básico sobre os principais tópicos abordados. Para isso será apresentado um breve resumo dos assuntos principais para entendimento geral do guia. É esperado que o leitor esteja familiarizado com grafos, linguagens de programação e teoria da computação.

Introdução

Segue-se uma introdução com os conceitos básicos para compreensão do processo para que possa ser possível o entendimento do guia e das expressões utilizadas. Os conceitos abortados são Sisal, IF1 e a Teoria dos Grafos.

Sisal e a IF1:

Sisal (*Streams and Iteration in a Single Assignment Language*) é uma linguagem de alto nível, de atribuição única e não puramente funcional, apresentando grande potencial de paralelismo[1]. Programas em Sisal são compilados para o formato intermediário e independente da máquina chamado IF1. Esse formato é então traduzido para grafo de fluxo de dados por um sistema de compilação.

O compilador Sisal gera um arquivo de saída no formato IF1, uma representação gráfica e independente de máquina usada por todas as implementações Sisal. É uma "linguagem de grafos com hierarquia" que descreve os grafos de fluxo de dados produzidos pelas funções Sisal.

Os nós IF1 são classificados em simples e compostos. Os nós simples representam os elementos básico da computação, enquanto os nós compostos representam as expressões estruturadas Sisal. Cada nó IF1 tem portas de entrada e de saída para onde os dados são enviados. Cada aresta conecta uma porta de saída de um nó a uma porta de entrada de outro nó. Os valores constantes são representados por um tipo especial de aresta denominada aresta literal, cuja origem não reside num nó e produz sempre um mesmo valor pré-definido[2].

Um grafo IF1 encapsula um bloco de computação, como por exemplo, uma função Sisal, podendo ser classificado como um grafo de função ou um subgrafo de um nó composto. O grafo de função é o mais alto nível da hierarquia, podendo ser formado por vários nós sucessivamente, estabelecendo - se uma hierarquia de grafos.

[1] http://www2.cmp.uea.ac.uk/~jrwg/Sisal/03.Expressions.html.

[2] https://www.academia.edu/36792188 /IF1-Viewer_A_Visual_Tool_for_Graphical_Display_and_Execution _of_SISAL_Programs

Teoria de Grafos:

A teoria dos grafos ou de grafos é um ramo da matemática que estuda as relações entre os objetos de um determinado conjunto[3]. Para tal são empregadas estruturas chamadas de grafos, G (V, E), onde V é um conjunto não vazio de objetos denominados vértices (ou nós) e E (do inglês *edges* - arestas) é um subconjunto de pares não ordenados de V.

Dependendo da aplicação, arestas podem ou não ter direção, pode ser permitido ou não arestas ligarem um vértice a ele próprio e vértices e/ou arestas podem ter um peso (numérico) associado. Se as arestas têm um sentido associado (indicado por uma seta na representação gráfica) temos um grafo orientado. Um grafo com um único vértice e sem arestas é conhecido como grafo trivial[4].

Processo de Criação:

Esse processo é descrito em 4 secções, começando com a interpretação dos arquivos IF1 e terminando com o processamento do grafo.

1. Interpretação do arquivo IF1

Nessa primeira fase, é onde se realiza a leitura e a interpretação do código IF1. Para isso, é necessário entender e distinguir cada letra e número dentro do arquivo. Os conceitos explicados serão apenas os necessários para a conclusão do processo

Em um arquivo IF1, temos em cada linha uma letra para representar o que essa linha significa. Na Figura 1, por exemplo, temos linhas começando com T, C$, I, X, G, N, E e L. As letras T são declarações de variáveis onde cada número representa o valor e o tipo que está sendo atribuído aquela variável. As linhas que se iniciam com a letra C$ são comentários. As letras I, X e G são representações de Grafos, ou seja, após essa letra tudo abaixo é para estar dentro de um grafo. Exemplo disso são as letras N (*Nodes*), E (*Edges*) e L (*Literal Edge*). Vide Figura 1:

[3] https://www.ensinoeinformacao.com/teoria-dos-grafos
[4] https://www.obm.org.br/content/uploads/2017/01/Nivel1_grafos_bruno.pdf

```
T 1 1 0    %na=Boolean
T 2 1 1    %na=Character
T 3 1 2    %na=Double
T 4 1 3    %na=Integer
T 5 1 4    %na=Null
T 6 1 5    %na=Real
T 7 1 6    %na=WildBasic
T 8 10
T 9 8   6       0
T 10 3  9       9
T 11 8  4       0
T 12 8  6       11
T 13 8  6       12
T 14 3  13      9
T 15 4  4
T 16 8  6       9
T 17 3  16      9
T 18 4  6
C$  C Faked IF1CHECK
C$  D Nodes are DFOrdered
C$  F Livermore Frontend   Version1.8
I       10      "sin"   %sl=10  %sf=simpson.sis
X       14      "Simpson"       %sl=15  %sf=simpson.sis
N 1     135     %sl=18  %sf=simpson.sis
E       0 2     1 1     6       %na=b  %mk=V   %sl=18
E       0 1     1 2     6       %na=a  %mk=V   %sl=18
N 2     151     %sl=18  %sf=simpson.sis
E       0 3     2 1     4       %na=n  %mk=V   %sl=18
N 3     141     %sl=19  %sf=simpson.sis
E       0 3     3 1     4       %na=n  %mk=V   %sl=19
L               3 2     4 "1"   %mk=V
```

Figura 1: Exemplo de um código IF1

Interpretando a Figura 1 temos:

- Declaração de variáveis (Linhas iniciadas com a Letra T);
- Comentários (Linhas iniciadas com a Letra C$);
- Inicialização do grafo (Linhas iniciadas com as letras I e X);
- Esse grafo é composto por 3 nós (Linhas iniciadas com a letra N);
- Possui também 4 arestas (Linhas Iniciadas com a Letra E);
- E apenas uma Aresta literal (Linhas Iniciadas com a Letra L);

Agora que já é possível identificar os componentes de um Arquivo IF1, é necessário entender como se caracteriza um nó e uma aresta. Cada nó pode ter diversas entradas, ou seja, várias arestas podem apontar para o mesmo nó, mas somente uma aresta pode sair de um nó. Conforme Figura 2:

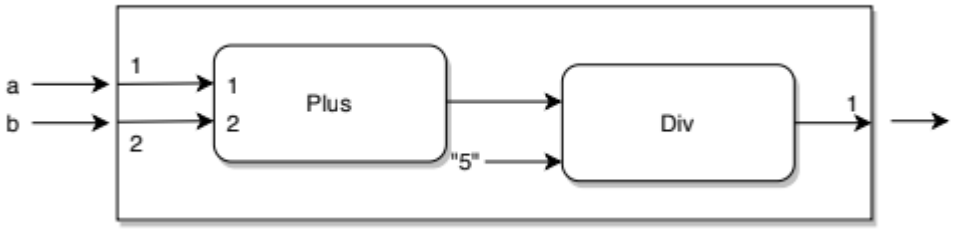


Figura 2: Grafo da Operação matemática (a+b)/5

É possível entender no exemplo da Figura 2 que cada nó tem diversas portas de entrada e apenas uma de saída e que cada nó tem sua função no processamento dos dados. No caso dos exemplos, os nós realizam a soma e a divisão. A inserção do número 5 no nó de divisão representa a Aresta Literal.

Para identificar as portas de entrada e saída de um nó no Arquivo IF1 é necessário se atentar aos números à direita de cada linha de um nó e/ou aresta. Utilizando o mesmo exemplo da Figura 1, temos os seguintes nós:

```
T 1 1 0    %na=Boolean
T 2 1 1    %na=Character
T 3 1 2    %na=Double
T 4 1 3    %na=Integer
T 5 1 4    %na=Null
T 6 1 5    %na=Real
T 7 1 6    %na=WildBasic
T 8 10
T 9 8   6       0
T 10 3  9       9
T 11 8  4       0
T 12 8  6       11
T 13 8  6       12
T 14 3  13      9
T 15 4  4
T 16 8  6       9
T 17 3  16      9
T 18 4  6
C$  C Faked IF1CHECK
C$  D Nodes are DFOrdered
C$  F Livermore Frontend   Version1.8
I       10      "sin"   %sl=10  %sf=simpson.sis
X       14      "Simpson"       %sl=15  %sf=simpson.sis
N 1     135     %sl=18  %sf=simpson.sis
E       0 2     1 1     6       %na=b  %mk=V   %sl=18
E       0 1     1 2     6       %na=a  %mk=V   %sl=18
N 2     151     %sl=18  %sf=simpson.sis
E       0 3     2 1     4       %na=n  %mk=V   %sl=18
N 3     141     %sl=19  %sf=simpson.sis
E       0 3     3 1     4       %na=n  %mk=V   %sl=19
L               3 2     4 "1"   %mk=V
```

Figura 3: Arquivo IF1 com os Nós destacados

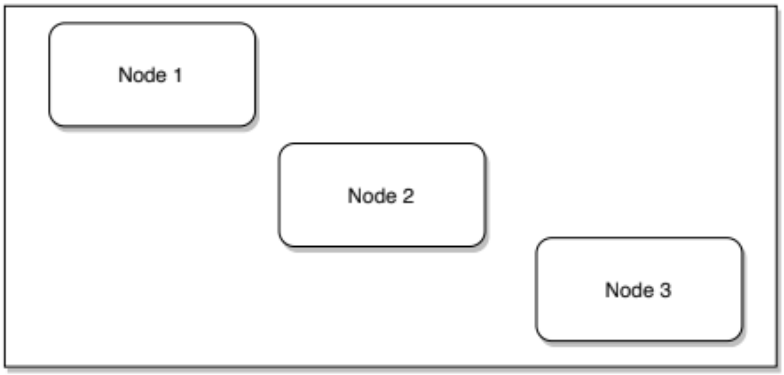


Figura 3: Representação dos nós da Figura 3

Cada aresta possui 4 números seguidos, sendo os dois primeiros o nó e a porta de origem e os dois últimos o nó e a porta de destino. Nesse caso temos a primeira Aresta saindo do nó 0 na porta 2 e o seu destino é o Nó 1 na porta 1 e o seu nome é “B” (definido pelos caracteres %na=b). Representando o exemplo da Figura 1 temos então:

```
T 1 1 0    %na=Boolean
T 2 1 1    %na=Character
T 3 1 2    %na=Double
T 4 1 3    %na=Integer
T 5 1 4    %na=Null
T 6 1 5    %na=Real
T 7 1 6    %na=WildBasic
T 8 10
T 9 8   6       0
T 10 3  9       9
T 11 8  4       0
T 12 8  6       11
T 13 8  6       12
T 14 3  13      9
T 15 4  4
T 16 8  6       9
T 17 3  16      9
T 18 4  6
C$  C Faked IF1CHECK
C$  D Nodes are DFOrdered
C$  F Livermore Frontend   Version1.8
I       10      "sin"   %sl=10  %sf=simpson.sis
X       14      "Simpson"       %sl=15  %sf=simpson.sis
N 1     135     %sl=18  %sf=simpson.sis
E       0 2     1 1     6       %na=b   %mk=V   %sl=18
E       0 1     1 2     6       %na=a   %mk=V   %sl=18
N 2     151     %sl=18  %sf=simpson.sis
E       0 3     2 1     4       %na=n   %mk=V   %sl=18
N 3     141     %sl=19  %sf=simpson.sis
E       0 3     3 1     4       %na=n   %mk=V   %sl=19
L               3 2     4 "1"   %mk=V
```

Figura 4: Arquivo IF1 com a Aresta sendo destacada

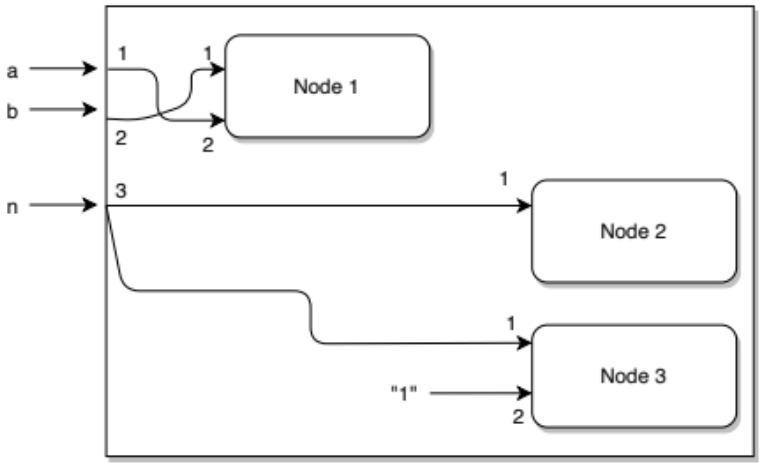


Figura 5: Representação de nós e Arestas da Figura 1

Nas Figuras 4 e 5, pode-se notar as portas de origem e destino de cada nó. A única que não possui uma porta e nó de origem é a Aresta Literal (representada na Figura 4 como “1”).

É possível encontrar subgrafos dentro dos arquivos IF1. Eles são representados pelo símbolo ‘{}’, significa que todos os nós e arestas dentro das chaves são subgrafos de um nó (Figura 6).

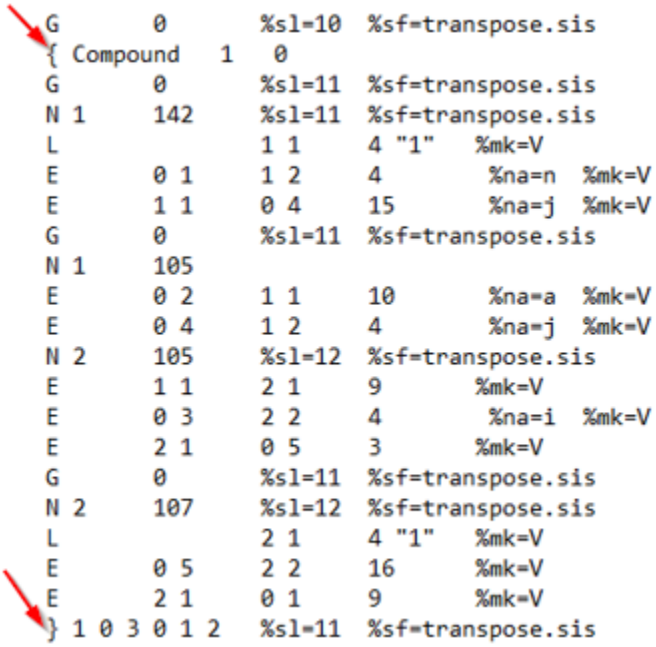


Figura 6: Exemplo de um Subgrafo em IF1.

Note que ao abrir Chave, todos os nós e arestas fazem parte de um subgrafo do Nó 1 que é representado ao fechar as chaves (Última linha na Figura 6).

2. Base de dados do diagrama na planilha

A ferramenta usada para processar o grafo permite que seja inserido uma planilha com as informações de nós e aresta para facilitar a geração do grafo. Assim, utilizando uma planilha em formato .csv é possível gerar o diagrama e ainda ter a base de dados necessária para ser importada pelo Gephi posteriormente.

O Excel tem diversos recursos e um deles é a manipulação de diagramas baseado numa fonte de Dados, sendo assim é possível obter os Diagramas e ao mesmo tempo a Base de Dados para depois der importada pelo Gephi. Veja na Figura 7.

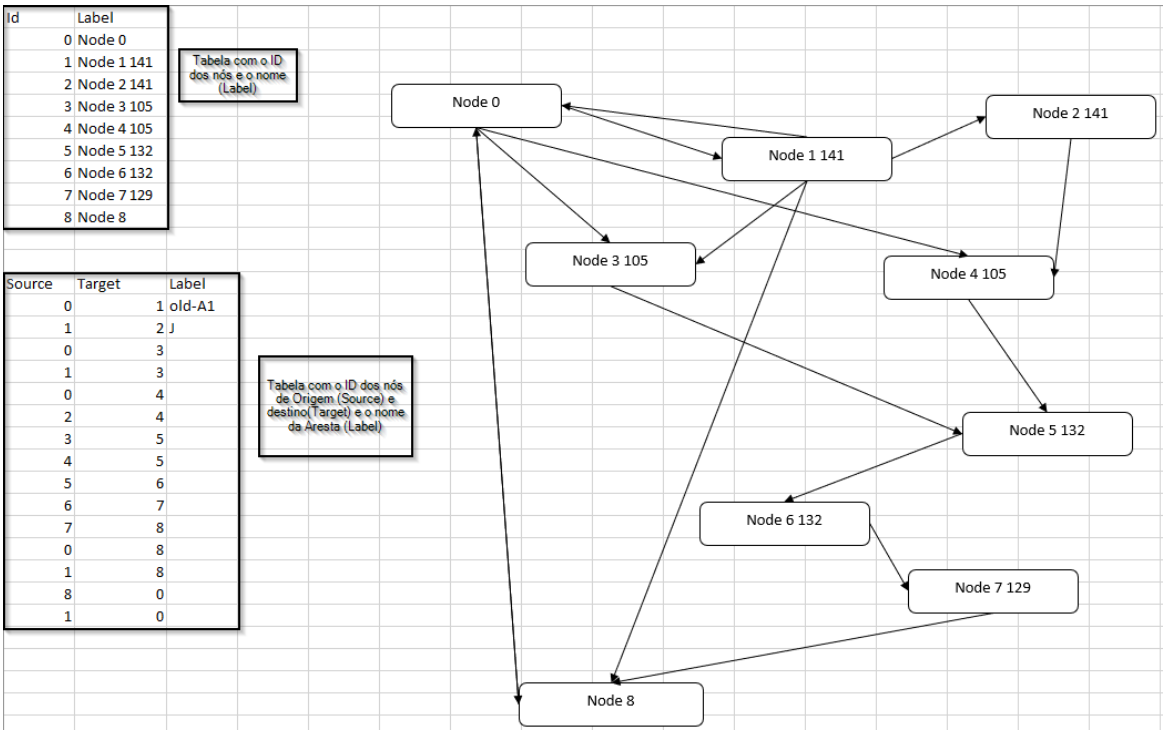


Figura 7: Exemplo de um grafo no Excel a partir de uma tabela com Dados

A Figura 7 foi retirada de uma planilha no Excel onde percebe-se que os nós são baseados na primeira tabela e as Aresta baseadas na segunda tabela. Com isso, na mesma plataforma é possível obter a imagem do Grafo e a base de Dados necessária para que o Gephi possa importa-la.

Para que seja possível exportar esses dados para o Gephi é mandatório que as colunas estejam como na Figura 7. A partir disso, copiar-se cada uma das tabelas para outro arquivo com extensão .csv, gerando então dois arquivos. Um com a tabela dos nós e outro com a das arestas.

3. Exportação dos Dados

A versão do Gephi utilizada nesse documento é a “gephi-0.9.2”, disponível para download gratuitamente (Linux e Windows). É considerado nesse capítulo que o programa já esteja instalado e configurado.

Ao criar um Projeto, na aba de Laboratório de Dados, deve-se importar primeiro a planilha dos nós e em seguida a da arestas.

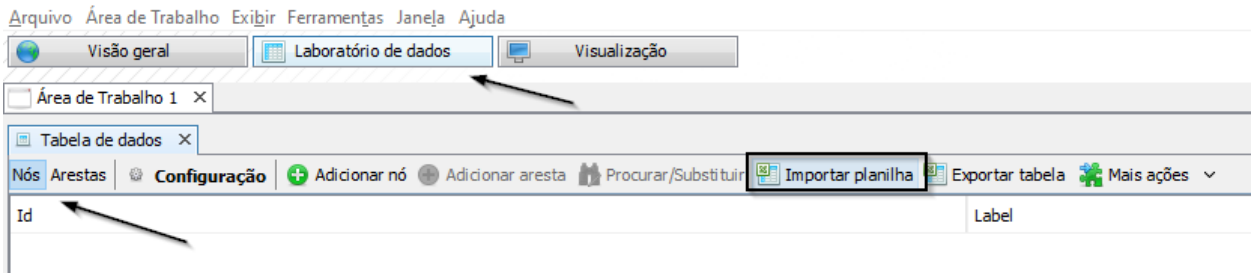


Figura 8: Imagem do Gephi para Importar uma Planilha

Ao selecionar o seu arquivo após ter clicado em “Importar Planilha”, será aberta uma tela para que seja selecionado o tipo de Separador, deve-se selecionar a opção *Semicolon* (ponto e vírgula). Em seguida, será apresentado uma previsão de como ficará sua tabela e caso esteja certo basta finalizar e já está pronto.

No caso da Aresta basta selecionar na Aba superior no lado esquerdo a opção “Aresta”, e repetir o mesmo processo. Porém agora deverá ser selecionado o arquivo .csv que contém a planilha das arestas.

Ao concluir a importação dos dois arquivos, clicando na Aba Visão Geral ao lado do Laboratório de Dados, será possível visualizar o grafo gerado a partir das informações inseridas anteriormente.

4. Processamento do grafo

Com o grafo já visível no Gephi, fica faltando apenas o seu processamento. O menu “Visão Geral” possui uma aba chamada Estatísticas, conforme Figura 9. A partir dela, é possível gerar os resultados do processamento do grafo clicando em Executar.

Para uma visão mais detalhada de cada um dos resultados, basta selecionar o ponto de Interrogação ao final da estatística desejada. Será aberto um gráfico mostrando os resultados obtidos e o porquê do valor final.

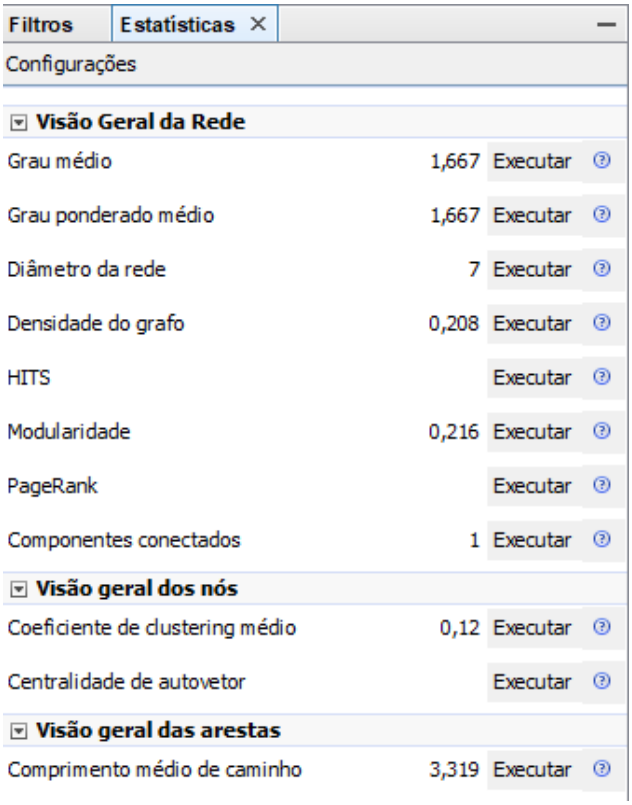
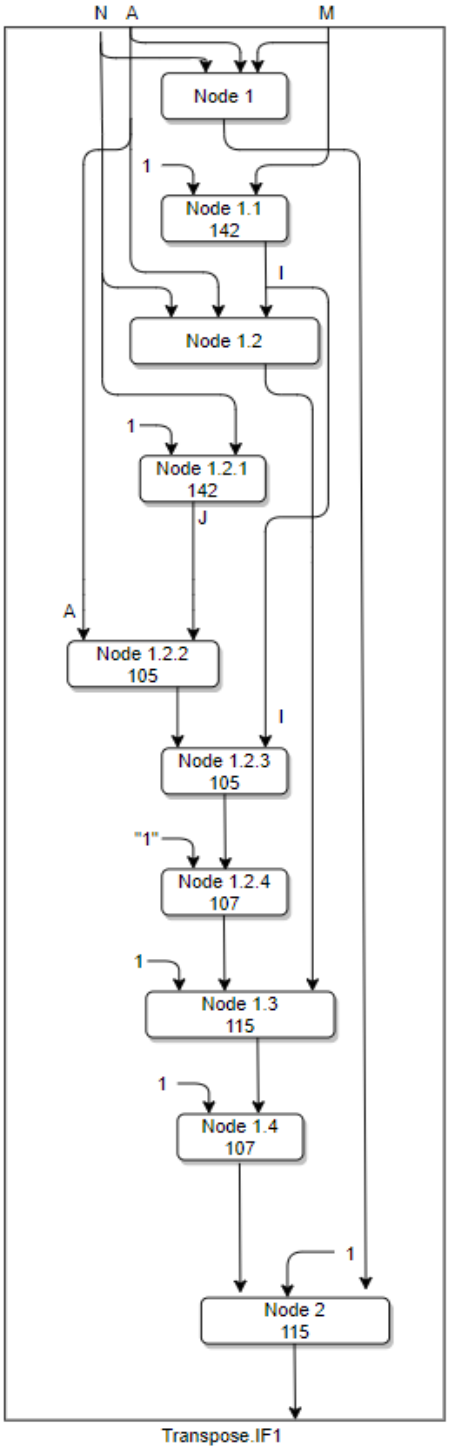


Figura 9: Exemplo de resultado obtido a partir do processamento do grafo

Passo a passo:

Interpretação do Arquivo IF1 e geração do Diagrama. Note que os nomes dos nós foram colocados em forma de secções para facilitar o entendimento dos subgrafos presentes.

```
C$  C Faked IF1CHECK
C$  D Nodes are DFOrdered
C$  F Livermore Frontend   Version1.8
X       14      "transpose"     %sl=8   %sf=transpose.si
{ Compound   1   0
G       0       %sl=10  %sf=transpose.sis
N 1     142     %sl=10  %sf=transpose.sis
L               1 1     4 "1"   %mk=V
E       0 1     1 2     4        %na=m  %mk=V   %sl=10
E       1 1     0 4     15       %na=i  %mk=V
G       0       %sl=10  %sf=transpose.sis
{ Compound   1   0
G       0       %sl=11  %sf=transpose.sis
N 1     142     %sl=11  %sf=transpose.sis
L               1 1     4 "1"   %mk=V
E       0 1     1 2     4        %na=n  %mk=V   %sl=10
E       1 1     0 4     15       %na=j  %mk=V
G       0       %sl=11  %sf=transpose.sis
N 1     105
E       0 2     1 1     10       %na=a  %mk=V
E       0 4     1 2     4        %na=j  %mk=V
N 2     105     %sl=12  %sf=transpose.sis
E       1 1     2 1     9        %mk=V
E       0 3     2 2     4        %na=i  %mk=V
E       2 1     0 5     3        %mk=V
G       0       %sl=11  %sf=transpose.sis
N 2     107     %sl=12  %sf=transpose.sis
L               2 1     4 "1"   %mk=V
E       0 5     2 2     16       %mk=V
E       2 1     0 1     9        %mk=V
} 1 0 3 0 1 2   %sl=11  %sf=transpose.sis
E       0 2     1 1     4        %na=n  %mk=V   %sl=11
E       0 3     1 2     10       %na=a  %mk=V   %sl=11
E       0 4     1 3     4        %na=i  %mk=V   %sl=11
N 2     115     %sl=12  %sf=transpose.sis
E       1 1     2 1     9        %mk=V
L               2 2     4 "1"   %mk=V
E       2 1     0 5     9        %mk=V
G       0       %sl=10  %sf=transpose.sis
N 1     107     %sl=12  %sf=transpose.sis
L               1 1     4 "1"   %mk=V
E       0 5     1 2     17       %mk=V
E       1 1     0 1     10       %mk=V
} 1 0 3 0 1 2   %sl=10  %sf=transpose.sis
E       0 2     1 1     4        %na=m  %mk=V   %sl=10
E       0 1     1 2     4        %na=n  %mk=V   %sl=10
E       0 3     1 3     10       %na=a  %mk=V   %sl=11
N 2     115     %sl=12  %sf=transpose.sis
E       1 1     2 1     10       %mk=V
L               2 2     4 "1"   %mk=V
E       2 1     0 1     10       %mk=V
```



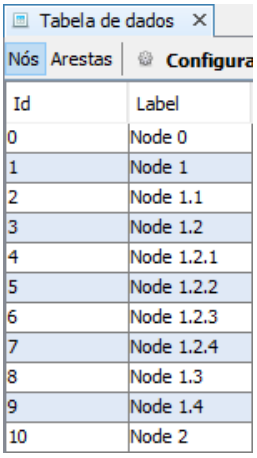
Transpose.IF1

Junto com o diagrama, necessita ser feita a base de dados em planilhas para ser importada pelo Gephi.

Source	Target	Label
0	1	N
0	1	A
0	1	M
0	2	M
0	3	N
0	3	A
2	3	I
0	4	N
0	5	A
4	5	J
5	6	

Id	Label
0	Node 0
1	Node 1
2	Node 1.1
3	Node 1.2
4	Node 1.2.1
5	Node 1.2.2
6	Node 1.2.3
7	Node 1.2.4
8	Node 1.3
9	Node 1.4
10	Node 2

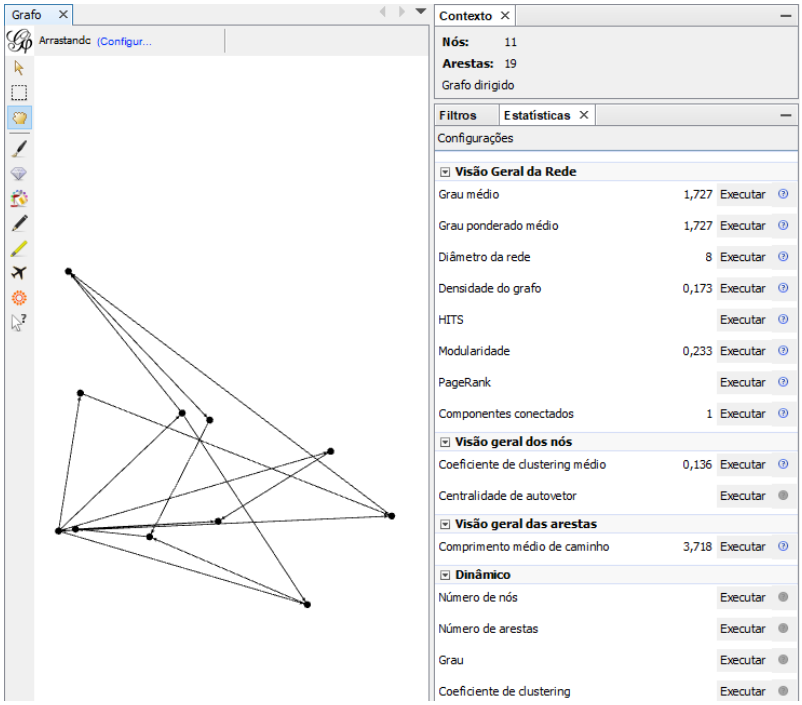
E em seguida exportar esses dados da planilha para o Gephi:



Id	Label
0	Node 0
1	Node 1
2	Node 1.1
3	Node 1.2
4	Node 1.2.1
5	Node 1.2.2
6	Node 1.2.3
7	Node 1.2.4
8	Node 1.3
9	Node 1.4
10	Node 2

Origem	Destino	Tipo	Tipo
0	1	Dirigido	N
0	1	Dirigido	A
0	1	Dirigido	M
0	2	Dirigido	M
0	3	Dirigido	N
0	3	Dirigido	A
2	3	Dirigido	I
0	4	Dirigido	N
0	5	Dirigido	A
4	5	Dirigido	J
5	6	Dirigido	
2	6	Dirigido	I
6	7	Dirigido	
7	8	Dirigido	
3	8	Dirigido	
8	9	Dirigido	
9	10	Dirigido	
1	10	Dirigido	
10	0	Dirigido	

Com as informaões já preenchidas no Laboratorio de Dados do Gephi, fica pendente apenas gerar o Grafo e processa-lo. O resultado do exemplo ficou assim:



Com o entendimento do código Sisal, a linguagem intermediaria IF1 e as propriedades da Teoria dos Grafos, foi possível realizar e gerar a tradução desses códigos a partir deste guia ilustrativo. Com o intuito de facilitar e ajudar em futuras pesquisas, ganhando tempo e otimizando o processo.

Esse guia está disponível nos seguintes links:

- https://github.com/RenanEug/Guia-Ilustrativo-para-convers-o-de-codigos-IF1-para-Grafos-na-Ferramenta-Gephi.git
- https://github.com/CafeForte/PublicDocs

Bibliografia

Skedzielewski S. and Glauert J. IF1 - An intermediate form for applicative languages. Manual M-170, Lawrence Livermore National Laboratory, Livermore, Calif., July 1985. Acesso em: 09/11/2019

H.B Chen, Behrooz Shirazi, and Susan Thrane.IF1-Viewer A Visual Tool for Graphical Display and Execution of SISAL Programs. University of Texas-Arlington Dept. of Computer Science and Engineering Arlington, Texas 76 109. Acesso em: 09/11/2019

Skedzielewski S. and Welcome M. Data-flow graph optimization in IF1. Proc. of FPCA'85, LNCS 201, 1985 Manual M-170, Lawrence Livermore National Laboratory, Livermore, Calif., July 1985. Acesso em: 09/11/2019.

Sisal Syntax: Expressions.
Disponivel em: http://www2.cmp.uea.ac.uk/~jrwg/Sisal/03.Expressions.html. Acesso em: 09/11/2019.

P. Feofiloff, Y. Kohayakawa, Y. Wakabayashi. Uma Introdução Sucinta à Teoria dos Grafos 12/07/2011.
Disponivel em: https://www.ime.usp.br/~yw/publications/books/TeoriaDosGrafos.pdf. Acesso em: 14/09/2019.

Feo J.T., Cann D.C., Oldehoeft R.R., ``A Report on the Sisal Language Project" Journal of
Parallel and Distributed Computing, December 1990.
Acesso em: 15/02/2020

J. R. Gurd P. M. C. C. Barahona A. P. W. BöhmC. C. KirkhamA. J. ParkerJ. Sargeant I. Watson. Fine-grain parallel computing: The dataflow approach 28 May 2005.
Acesso em: 22/02/2020

Skedzielewski, S.K. Welcome ML Data Flow Graph Otimization in IF1 in Functional Programming Languages and Computer Architeture.
Acesso em: 22/02/2020