CENTRO DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE FEDERAL DA PARAIBA

SIMIN JALALI R. RABBANI

ARRANJOS ÓTIMOS DE DESLOCAMENTO DE EQUIPAMENTO ENTRE DIFERENTES POLOS

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS CURSOS DE PÓS GRADUAÇÃO DE ENGENHARIA DO CENTRO DE CIÊNCIAS E TECNOLOGIA DA UNIVERSIDADE FEDERAL DA PARAÍBA COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS (M.Sc.).

COMISSÃO EXAMINADORA:

JEAN CLAUDE PICARD - Ph.D.
Presidente

EDUARDO ANDRADE VELOSO - M.Sc.
Examinador Interno

MARCOS A. ZAMPIERI NUNES - M.Sc.
Examinador Externo

Campina Grande - Paraiba
Agosto - 1977

R113a Rabbani, Simin Jalali R.
Arranjos ótimos de deslocamento de equipamento entre diferentes polos / Simin Jalali R. Rabbani. - Campina Grande, 1977.
103 f.

Dissertação (Mestrado em Ciências) - Universidade Federal da Paraíba, Centro de Ciências e Tecnologia, 1977.
"Orientação : Prof. Dr. Jean Claude Picard".
Referências.

1. Transportadores de Equipamentos - Arranjos entre Polos. 2. Deslocamento de Equipamentos. 3. Arranjos de Deslocamento. 4. Dissertação - Ciências. I. Picard, Jean Claude. II. Universidade Federal da Paraíba - Campina Grande (PB). III. Título

CDU 621.87:519.14(043)

Aos meus pais

AGRADECIMENTOS

Ao seu orientador Prof. Jean Claude Picard pela assistência dada durante a execução deste trabalho, sem a qual não seria possível sua conclusão.

Ao Prof. Marcos A. Zampieri Nunes, Prof.Luiz Carlos Marcondes e Prof. Eduardo Andrade Veloso que de uma forma ou de outra, prestaram sua colaboração.

A Senhora Leônia Leão da Nóbrega por sua ajuda na preparação dessa dissertação.

A todos os colegas de curso pelo apoio sempre patenteado.

Finalmente (que não é de menor importância) agradece à D. Safa, Soheil, Massoud e Emília pela compreensão e paciência que sempre demonstraram, sem as quais não seria possível a realização deste trabalho.

A B S T R A C T

This work aims at the optimization of total travel time for transportation of equipments between different points. For the solution, a network of points in the North - East of Brazil is considered.

Three different methods using Graph Theory have been used to determine the optimal arrangement. The minimum cost movements are computed by using the algorithms of Klein; Busacker and Gowen; as well as the method of optimal assignment. Further the adequacy of the Clarke and Wright algorithm for solving such a problem is discussed subject to a few restrictions.

The results obtained in this study show that the above methods of graph theory can be used to determine the minimum travel time between different points.

R E S U M O

Este trabalho visa a otimização do tempo total de viagem, para o deslocamento de equipamentos entre diferentes polos. Para solucionar o problema utilizou-se uma rede de pontos existentes na região do Nordeste Brasileiro.

Três métodos diferentes empregando Teoria dos Grafos vem sendo utilizados para determinar o arranjo ótimo. Os movimentos ao custo mínimo são computados, emprega-se os métodos de Klein; Busacker e Gowen; e o da designação ótima. A adequação do método de Clarke e Wright para solucionar tal problema é discutida sujeito à algumas restrições.

Os resultados obtidos nesse estudo mostram que os métodos, referidos acima, podem ser utilizados para determinar o tempo mínimo de viagem entre diferentes polos.

Í N D I C E

CAPÍTULO I

I N T R O D U Ç Ã O

Devido a sua vasta aplicabilidade, o estudo da teoria dos grafos se expandiu rapidamente. Um fato importante nesse desenvolvimento foi o aperfeiçoamento dos computadores.

A representação detalhada e direta dos sistemas práticos, como as distribuições ou redes dos meios de comunicações conduz aos grafos de grandes tamanhos. O sucesso das análises desses grafos depende tanto da existência de bons algoritmos, como da disponibilidade de computadores velozes.

Neste trabalho se aplica a teoria dos grafos a fim de minimizar o tempo de deslocamento de equipamentos entre diversos polos.

No sentido de solucionar o referido problema, utilizou-se tres métodos teóricos de grafos; método de Busacker e Gowen, e de Klein e o método de designação.

Os métodos foram ensaiados satisfatoriamente sobre dados práticos.

A tese começa - mais particularmente - no Capítulo II com a definição do escopo deste trabalho. No Capítulo III serão explicados os conceitos gerais de teoria dos grafos necessárias ao bom entendimento das técnicas utilizadas neste trabalho. O Capítulo IV descreve o processo de coletar os dados. No Capítulo V serão apresentados os algoritmos utilizados. O Capítulo VI apresenta as técnicas utilizadas para resolver o problema, e os resultados são descritos no Capítulo VII.A conclusão do trabalho será apresentado no Capítulo VIII.

Os programas principais são encontrados no apêndice.

CAPÍTULO II

O B J E T I V O

Desenvolveu-se na Associação Técnico-Científica "Ernesto Luiz de Oliveira Júnior" - ATECEL, Órgão da Universidade Federal da Paraíba, um projeto denominado de "Pesquisa de Origem e Destino", no ano de 1975. A finalidade da pesquisa era obter um conjunto de informações básicas sobre o fluxo de riquezas escoadas ou recebidas na região Nordeste brasileira, identificando os polos de origem-destino, a freqüência deste intercâmbio, bem como da distância em que essas riquezas se deslocam para alcançarem o objetivo da comercialização. Para obter os objetivos definidos foram selecionadas (entre as vias de comunicações que compõem a malha rodoviária nordestina) as vias com valores significativos de volume de tráfego e alocou-se estrategicamente noventa e dois "Postos de Coleta de Dados", capazes de captar o intercâmbio entre os diversos polos internos na região nordestina e o restante do País.

Para a execução da coleta dos dados, o Projeto foi dividido em seis etapas de execução. Baseado no volume de tráfego existente no trecho escolhido, e as dificuldades de cada área de atuação. Para investigação em cada posto foi necessário de 1 a 2 conjuntos de equipamentos. O número máximo de postos investigados foi de 23 e, utilizou-se no máximo 25 conjuntos de equipamentos. (12)

O objetivo deste trabalho é o de se chegar ao arranjo "ótimo" de distribuição de equipamento, de maneira que, minimize o tempo de deslocamento ou, em outras palavras, os custos de transportes.

CAPÍTULO III

DEFINIÇÃO E REVISÃO DE CONCEITOS GERAIS

Neste capítulo serão apresentados apenas os conceitos gerais, e definições indispensáveis ao bom entendimento das técnicas utilizas nesse trabalho, e está basicamente dividido em tres partes. Na primeira é feita uma revisão dos conceitos mais utilizados da teoria dos grafos. Na segunda é estudado o problema de fluxo máximo em uma rede e o fluxo com o custo mínimo, e na terceira, são apresentados os problemas de atribuição, transporte e designação.

III.1 - CONCEITOS GERAIS

III.1.1 - GRAFO

Um grafo G é uma coleção de nós ou vértices

x_1, x_2, ... x_n (o conjunto X) e uma coleção de linhas a_1, a_2, ... a_m (o conjunto A) ligando todos, ou alguns desses vértices. Geralmente representa-se um grafo G por G(X,A).

Se os elementos em A tiverem uma direção, esta será usualmente indicada por uma seta, e são chamados arcos, e o grafo resultante é chamado de um grafo direcionado (Figura III.1.a). Se os elementos no conjunto A não tiverem nenhuma direção, são chamados ramos, e o grafo é um grafo não direcionado (Figura III.1.b). Um grafo mixto é um grafo que tem arcos e ramos (Figura III.1.c). Explica-se que um ramo em um grafo não direcionado ou mixto é equivalente a dois arcos em sentido contrário.

Uma maneira alternativa e freqüentemente preferível para descrever um grafo G, é especificado o conjunto X de vértices e uma correspondência Γ, a qual mostra como eles são interligados. Γ é chamada uma aplicação do conjunto X em X, e o grafo é denotado por G(X, Γ). No exemplo da Figura III.1.a tem-se:

$\Gamma(x_1) = \{x_2, x_5\}$; isto é, x_2, x_5 são os vértices finais dos arcos cujo nó inicial é x_1.

$\Gamma(x_2) = \{x_1, x_3\}$;

$\Gamma(x_3) = \{x_1\}$;

$\Gamma(x_4) = \phi$, o conjunto nulo, ou conjunto vazio

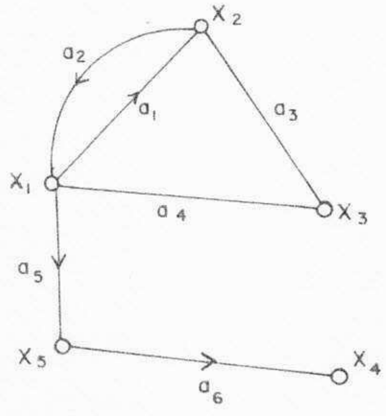


FIGURA III . 1.a. — GRAFO DIRECIONADO

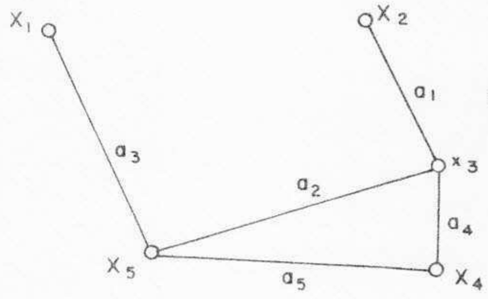


FIGURA III . 1.b - GRAFO NÃO - DIRECIONADO

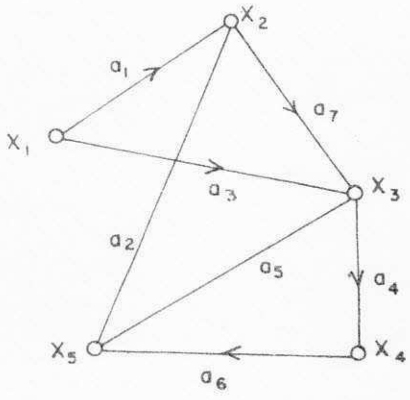


FIGURA III . 1.c - GRAFO MIXTO

De um modo simular o conjunto de vértices x_K para o qual um arco (x_K, x_i) existe no grafo, escreve-se como $\Gamma^{-1}(x_i)$. Portanto, a relação $\Gamma^{-1}(x_i)$ é chamada aplicação inversa. Por exemplo: para o grafo da Figura III.1.a tem-se:

$$\Gamma^{-1}(x_1) = \{x_2, x_3\}$$

$$\Gamma^{-1}(x_2) = \{x_1\}$$

Grafo Bipartido - Um grafo é chamado de bipartido se os vértices poderem ser particionados em dois conjuntos

$$S = \{s_i\} \ (i = 1, \ldots, m) \text{ e } T = \{t_j\} \ (j = 1, \ldots, n)$$

de maneira que todos os arcos sejam de tipo (s_i, t_j) (veja Figura III.4).

III.1.2 - CAMINHO

Em um grafo direcionado, uma seqüência de arcos onde o nó final de um arco é o vértice inicial do seguinte, é chamado de um caminho. Portanto, na Figura III.2 a seqüência dos vértices são caminhos:

$x_1, x_2, x_5, x_4, x_3, x_5, x_6$;

x_1, x_2, x_5, x_4, x_3;

$x_1, x_2, x_5, x_4, x_3, x_2, x_5, x_6$

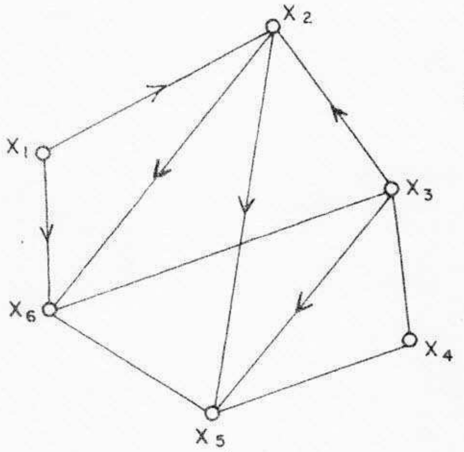


FIGURA III.2 - GRAFO PARA APRESENTAÇÃO CAMINHO

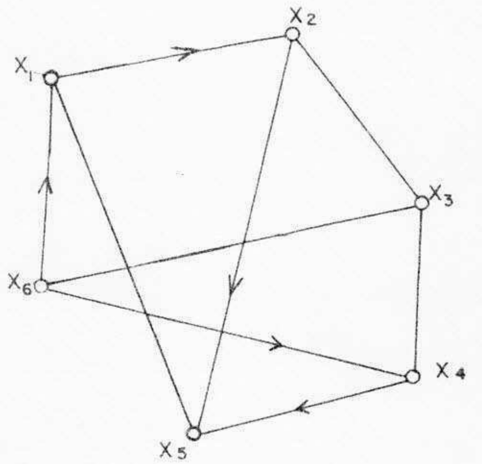


FIGURA III.3 - GRAFO PARA APRESENTAÇÃO CIRCUITO

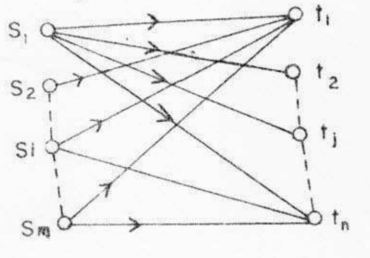


FIGURA III.4 - GRAFO BIPARTIDO

III.1.3 - CUSTO

Um número C_{ij} às vezes pode ser associado a um arco (x_i, x_j). Esses números são chamados de custos ou pesos, e, por isso, o grafo é chamado de um grafo com arcos ponderados.

Desde que pode se apresentar o caminho pela sequência dos arcos $(a_1, a_2, ..., a_q)$ o custo do caminho $L(\mu)$ é o somatório dos custos nos arcos que aparecem em μ, isto é:

$$L(\mu) = \Sigma C_{ij} (x_i, x_j) \text{ em } \mu$$

Cardinalidade do caminho μ é o número de arcos (q) que aparecem neste mesmo caminho .

III.1.4 - CIRCUITO

Um Circuito é um caminho $x_1, x_2, ..., x_n$ no qual o vértice inicial x_1 coincide com o vértice final x_n. Portanto, na Figura III.3 as seguintes seqüências formam circuitos:

x_1, x_2, x_5, x_1;

$x_5, x_1, x_2, x_3, x_6, x_1, x_2, x_5$

$x_1, x_2, x_3, x_6, x_4, x_5, x_1$

O problema do menor caminho é o de encontrar o caminho do valor mínimo de um vértice inicial $s \in X$ a um vértice final específico $t \in X$. Para cada grafo $G(X, \Gamma)$ com arcos ponderados, define-se uma matriz de custo $C = [C_{ij}]$. Os elementos da matriz de custo podem ser positivos, negativos ou nulos.

III.2 - REDES, FLUXO MÁXIMO E FLUXO DE CUSTO MÍNIMO

III.2.1 - REDE

Uma rede é um grafo $G(X, A)$ onde se associa para cada elemento do conjunto A um número b_{ij} denominado de capacidade do arco. Os elementos do conjunto de capacidade B podem variar de zero a infinito (∞). Existem dois vértices especiais em uma rede: a fonte (s) e o vértice terminal (t).

III.2.2 - FLUXO EM UMA REDE

Um conjunto de números f_{ij} é chamado de fluxo em uma rede, se satisfaz as seguintes restrições:

$$(1) \qquad \sum_{x_j \in \Gamma(x_i)} f_{ij} - \sum_{x_k \in \Gamma^{-1}(x_i)} f_{ki} = \begin{cases} V \text{ se } x_i = s \\ 0 \text{ se } x_i \neq s \text{ ou } t \\ -V \text{ se } x_i = t \end{cases}$$

(2) $0 \leq f_{ij} \leq b_{ij}$ $\forall (x_i, x_j) \in A$

O (V) que aparece na equação (1) é um número não negativo chamado de valor de fluxo.

Restrições (1) estabelecem que o fluxo que entra no vértice x_i é igual ao fluxo que sai do mesmo vértice. Para os vértices s e t o valor total de fluxo que sai do vértice s é igual a (V) e é, também, o valor de fluxo que chega ao vértice t.

Restrições (2) estabelecem que o fluxo f_{ij} no arco (x_i, x_j) é sempre menor ou igual a capacidade b_{ij} do arco.

Se se deseja obter o fluxo máximo numa rede, procura-se um conjunto de fluxos de modo que:

$$V = \sum_{x_j \in \Gamma(s)} f_{sj} = \sum_{x_k \in \Gamma^{-1}(t)} f_{kt}$$

seja máximo.

Encontrar o valor de fluxo máximo em qualquer rede é um problema de programação linear com a função objetiva (a ser maximizada) z = V e as restrições (1) e (2). Desde que este é um caso muito especial de programação linear, pode ser resolvido com algoritmos mais eficientes do que o método simplex. Um deles muito conhecido e eficiente é o algaritmo de FORD e FULKERSON (9).

Numa rede pode existir n_s origens e n_t destinos; supõe-se que o fluxo pode ser transferido de qualquer fonte para qualquer destino. Este problema pode ser transformado em uma rede com um vértice inicial e outro terminal.

Ilustração - Determine o fluxo máximo de um vértice inicial s a um vértice final t na rede da Figura III.4. Supõe-se que a capacidade nos arcos n_s, T_j seja igual a infinito (∞).

O primeiro número associado aos arcos é o da capacidade (b_{ij}) e o segundo é o do fluxo no arco (f_{ij}). Verifica-se que este Fluxo tem um valor de 6.

III.2.3 - FLUXO COM CUSTO MÍNIMO EM UMA REDE

Na Secção anterior (III.2.2), um número chamado capacidade foi associado a cada arco. Agora será associado, também, um outro número a cada arco, que é o custo C_{ij} necessário para transportar uma unidade de fluxo ao longo do arco (x_i, x_j).

O interesse é de transportar um fluxo de valor V de s para t, ao custo mínimo. Se não houvesse a restrição de capacidade, o problema se tornaria o de achar o menor caminho de s para t, e transportar todo o fluxo ao longo desse caminho. É obvio que, o valor de fluxo desejado V não deve ser maior que o do fluxo máximo de s para t, senão, não existiria nenhuma solução. No caso em que o valor de fluxo

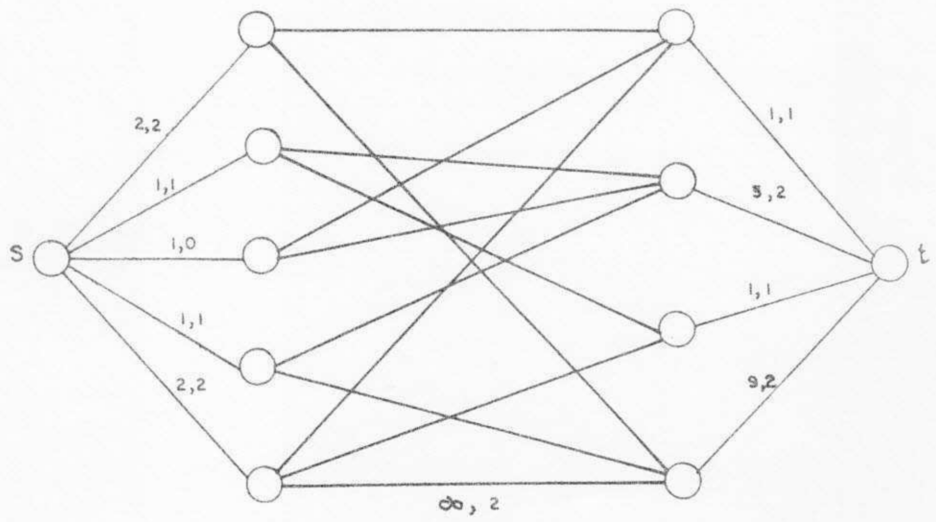


FIGURA III.4 — FLUXO MÁXIMO NUMA RÊDE

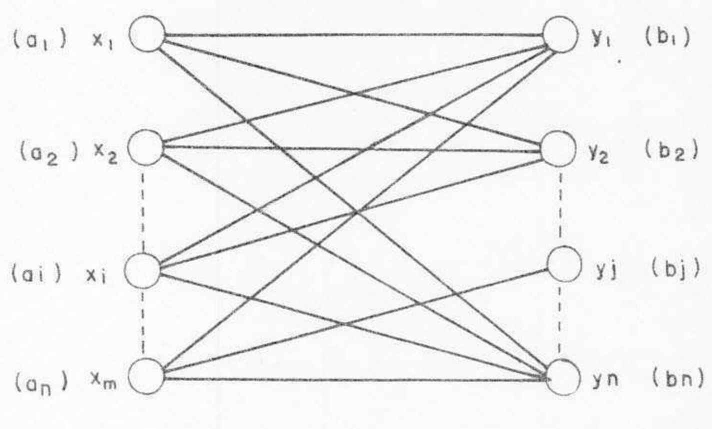


FIGURA III.5 — UMA RÊDE BIPARTIDA

(V) seja menor ou igual ao valor máximo possível, pode-se obter diferentes soluções ótimas.

Portanto, deseja-se transportar um fluxo de valor V com o custo mínimo, ou seja:

$$\text{Min: } z = \sum_{(x_i, x_j) \in A} \sum a_{ij} (f_{ij})$$

sujeito à:

$$\sum_j f_{ij} - \sum_k f_{ki} = \begin{cases} V \text{ se } i = s \\ 0 \text{ se } i \neq s \text{ ou } t \\ -V \text{ se } i = t \end{cases}$$

$$0 \leq f_{ij} \leq b_{ij} \; \forall (i, j) \in A$$

III.3 - PROBLEMA DE TRANSPORTE

O problema padrão de transporte é aquele que, às vezes, se refere como problema de Hitchcock, que foi um dos seus formuladores. O estabelecimento do problema de Hitchcock que é um caso particular do problema de fluxo com custo mínimo definido na Secção III.2.3, é o seguinte:

Suponha que existem m origens $x_1, x_2, \ldots, x_m$ para uma comodidade, com a (x_i) unidades de suplementos em

x_i e n destinos $y_1, y_2, \ldots, y_n$ para a comodidade, com uma demanda $b(y_j)$ em y_j. Seja a_{ij} o custo para transportar uma unidade de x_i para y_j, o problema é determinar o fluxo que satisfaz às demandas e minimiza o custo de transporte. O problema é equivalente a:

$$\text{Min: } z = \sum_{i=1}^{m} \sum_{j=1}^{n} a_{ij} f_{ij}$$

$$\sum_{j=1}^{n} f_{ij} \leq a_i \qquad (i = 1, 2, \ldots, m)$$

$$\sum_{i=1}^{n} f_{ij} \geq b_j \qquad (j = 1, 2, \ldots, n)$$

$$f_{ij} \geq 0$$

Ou a achar um fluxo de valor $\sum_{j=1}^{n} b_j$ e de custo mínimo em uma rede bipartida. (Figura III.6).

Supondo-se que $\sum_{i=1}^{m} a_i \geq \sum_{j=1}^{n} b_j$ e que a_i, b_j são inteiros (equivalentemente, presume-se que são racionais).

III.3.1- PROBLEMA DE DESIGNAÇÃO ÓTIMA

O problema de designação ótima pode ser considerado como um caso especial do problema de Hitchcock ou seja:

- número de origens = número de destinos(m=n)

- capacidade de cada origem = 1 ($a_i = 1, \ldots, m$);
- demanda de cada destino = 1 ($b_j = 1, \ldots, n$).

Obter-se-á o modelo da designação que tem o seguinte aspecto:

$$\text{Min:} \quad \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} \, f_{ij}$$

$$\sum_{j=1}^{n} f_{ij} = 1 \quad (i = 1, 2, \ldots, n)$$

$$\sum_{i=1}^{n} f_{ij} = 1 \quad (j = 1, 2, \ldots, n)$$

$$f_{ij} \geq 0 \quad (i = 1, 2, \ldots, n), (j = 1, 2, \ldots, n)$$

CAPÍTULO IV

COLETA DE DADOS

IV. 1 - INTRODUÇÃO

Como foi visto no Capítulo II, os dados da pesquisa de (O-D) do Nordeste Brasileiro foram obtidos num trabalho realizado em 6 etapas, sendo que, em cada etapa, foram alocados no máximo 23 Postos, cada Posto com 1 a 2 conjuntos de equipamentos. Utilizou-se no máximo um total de 25 conjuntos de equipamentos. A relação das equipes, etapas e Postos de Pesquisa na Coleta de Dados é apresentada na Tabela IV.1.

Para otimização do tempo de deslocamento de equipamentos entre Postos das diferentes etapas, os dados necessários são os tempos de viagem de cada Posto em cada etapa, e todos os Postos da etapa seguinte.

GRUPOS DE PESQ.	1a ETAPA		2a ETAPA		3a ETAPA		4a ETAPA		5a ETAPA		6a ETAPA	
	POSTO	EST.	POSTO	EST.	POSTO	EST.	POSTO	EST.	POSTO	EST.	POSTO	EST.
1	01 02	MA			09 10	PI	11 12	PI	46 47	PE	60	
2	07 08	MA	04	MA	05 06	MA	03	MA	31 32	RN		
3	13 14	PI	15	CE			16 17	CE	18 19	CE	20	CE
4					21 22	CE	23 24	CE	25 40	CE PB	37 39	PB
5			42 43	PE	26 27	CE	30 45	RN PE	36 38	PB PB	64 66	AL SE
6			56+	PE	57+	PE			58+	PE	59+	PE
7					51 61	PE AL	54 55		44 48		63 67	AL SE
8					50 62	PE AL	49	PE				
9					41 69	PE BA	73 76	BA	28 29	RN	33 34	RN PB
10			35+	PB	78+	BA	83+	BA	77+	BA	91 92	MG
11					84 86	BA	79 80	BA	65 68	SE	87 90	BA
12					89	BA	88	BA	85	BA	70	BA
13					72	BA	71	BA	81 82	BA	74 75	BA
14									52 53	PE		

TABELA IV.1 - RELAÇÃO DAS EQUIPES, ETAPAS E PÔSTOS DE PESQUISA NA COLETA DE DADOS

os (+) significa que no posto referente precisa-se de dois conjuntos de equipamentos.

A Secção seguinte apresenta os passos do processos de arranjo de dados.

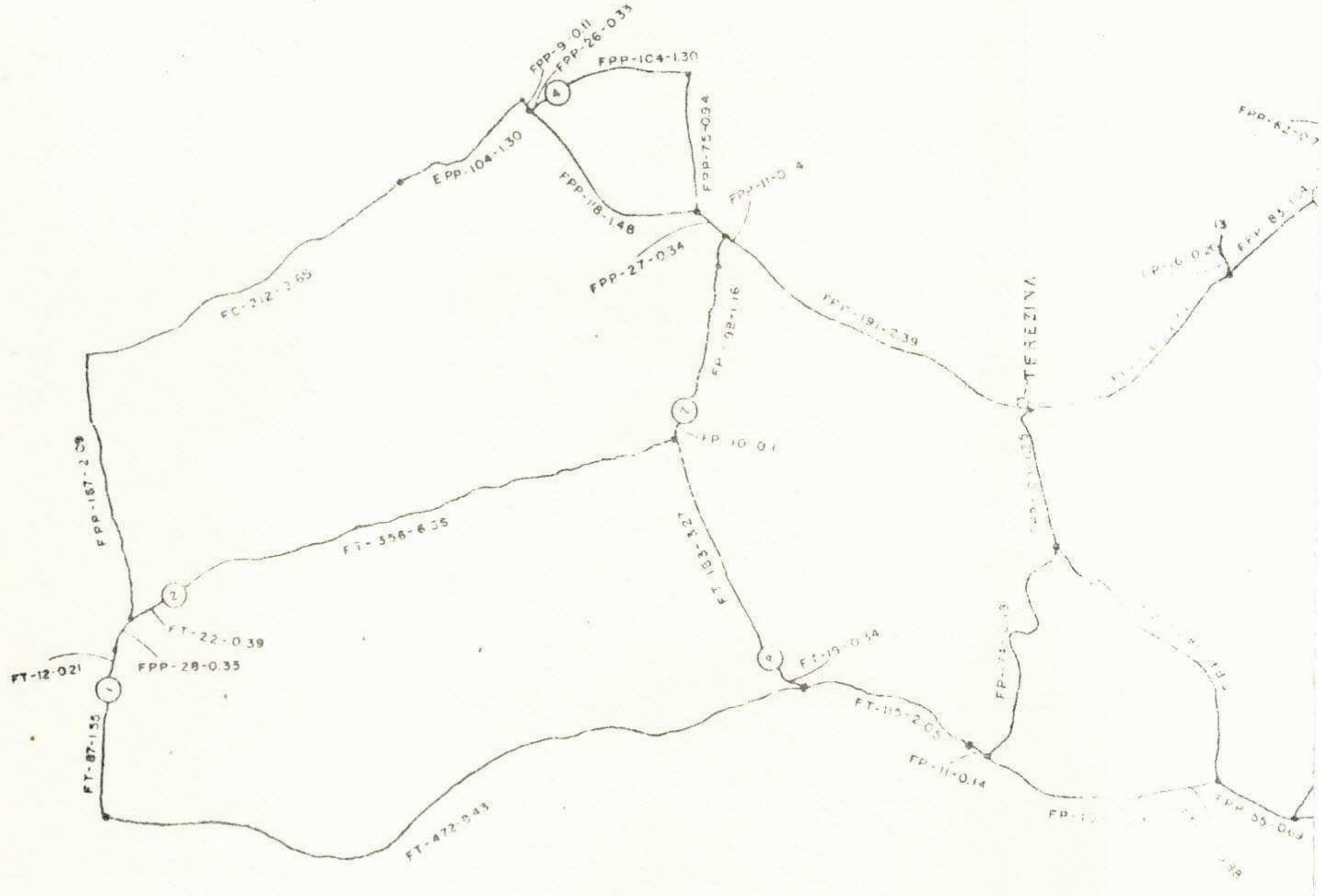
IV.2 - ORGANIZAÇÃO DE DADOS

Para se manipular os dados, inicialmente os Postos de Coleta de Dados foram marcados no mapa do Nordeste Brasileiro, e procedeu-se ao estudo nos passos seguintes:

1º PASSO - Preparou-se as redes viárias, interligando os Postos de cada etapa, por todas as estradas transitáveis, aos postos da etapa seguinte. Estas redes viárias são representadas nos grafos, e mostrados nas figuras (IV.1) a (IV.5).

2º PASSO - Transferiu-se para os gráficos a distância entre todos os pares de vértices adjacentes. (Vértices são os Postos ou pontos de interseção das estradas). Convencionalmente, nos gráficos, foram mostrados os diferentes tipos de estradas (*)

(*) FPP - Estrada Pavimentada Federal
EPP - Estrada Pavimentada Estadual
FP - Estrada em Pavimentação Federal
EP - Estrada em Pavimentação Estadual
FT - Estrada de Terra Federal
ET - Estrada de Terra Estadual



FPP-9-0.11
FPP-26-0.33
FPP-104-1.30
FPP-75-0.94
FPP-118-1.48
FPP-27-0.34
FPP-191-2.39
TEREZINA
FT-358-6.35
FT-22-0.39
FPP-28-0.35
FT-12-0.21
FT-87-1.55
FT-472-9.43
FT-19-0.34
FT-115-2.05
FP-11-0.14
FPP-55-0.09

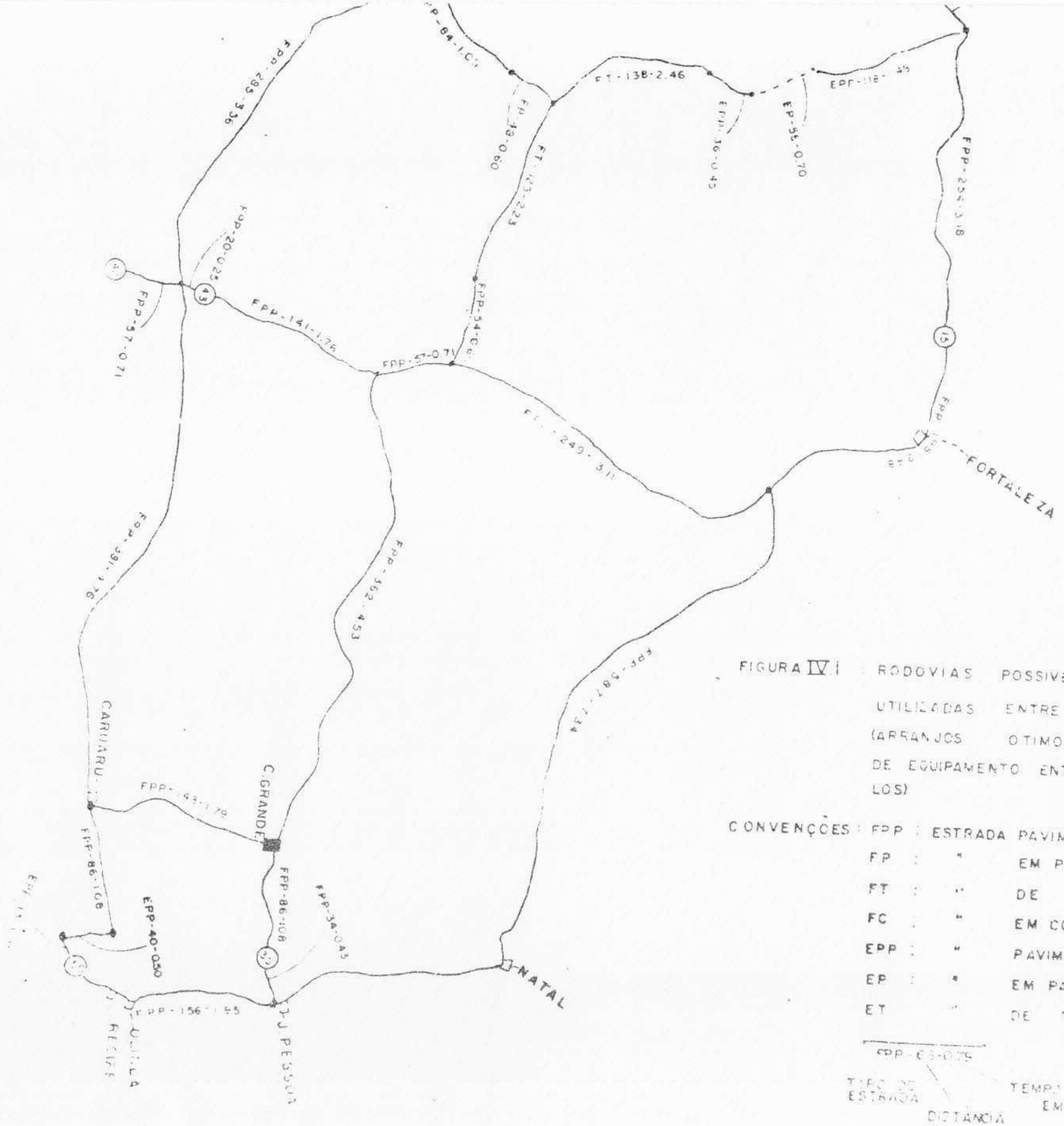


FIGURA IV.1 : RODOVIAS POSSIVEIS DE SEREM UTILIZADAS ENTRE AS ETAPAS 1 e 2 (ARRANJOS OTIMOS DE DESLOCAMENTO DE EQUIPAMENTO ENTRE DIFERENTES PO LOS)

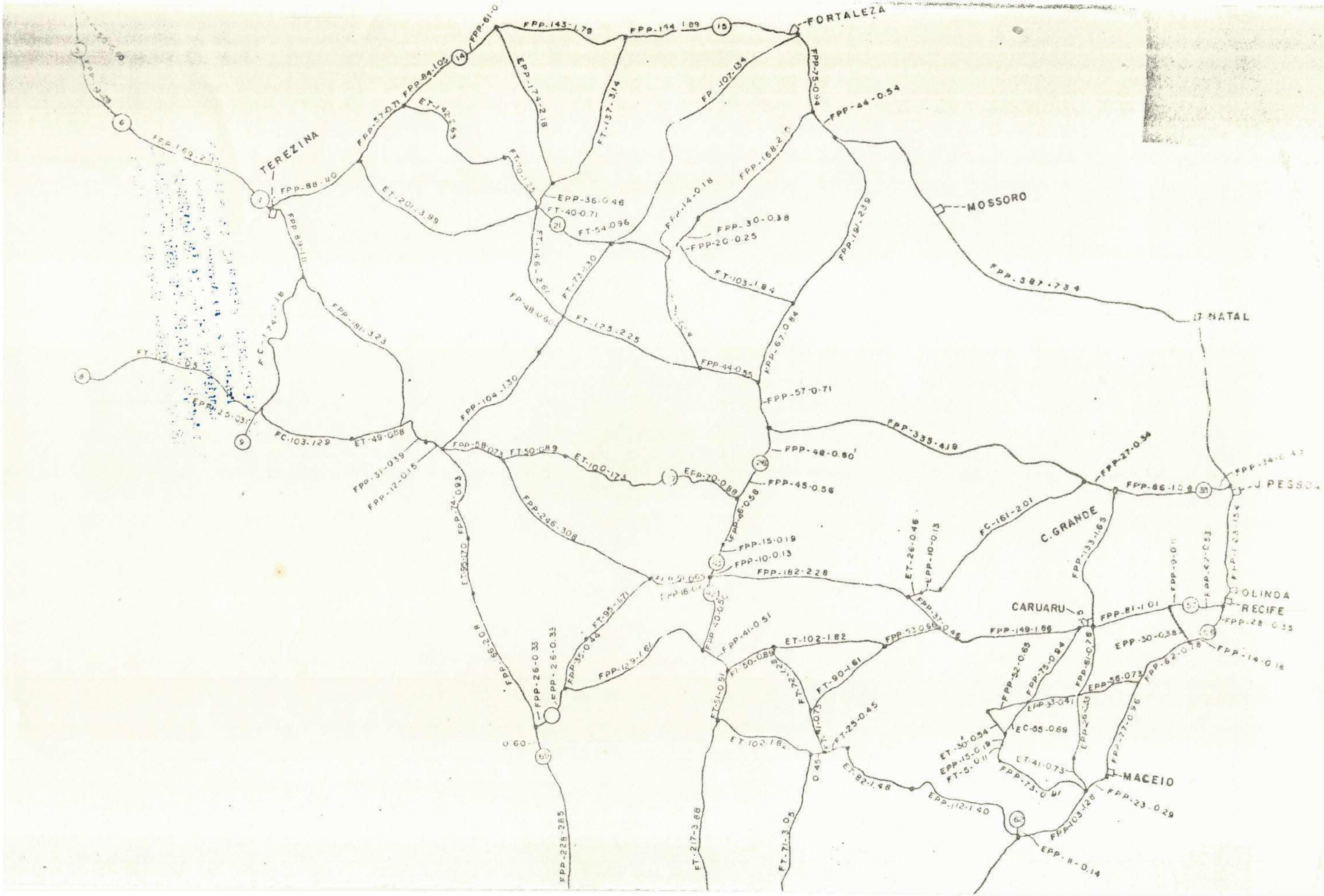
CONVENÇÕES : FPP : ESTRADA PAVIMENTADA FEDERAL
FP : " EM PAVIMENTAÇÃO FEDERAL
FT : " DE TERRA FEDERAL
FC : " EM CONSTRUÇÃO FEDERAL
EPP : " PAVIMENTADA ESTADUAL
EP : " EM PAVIMENTAÇÃO ESTADUAL
ET : " DE TERRA ESTADUAL

FPP-63-0.75

TIPO DE ESTRADA

DISTANCIA EM KM

TEMPO DE VIAGEM EM HORA


FORTALEZA
TEREZINA
MOSSORO
NATAL
J. PESSOA
C. GRANDE
CARUARU
OLINDA
RECIFE
MACEIO

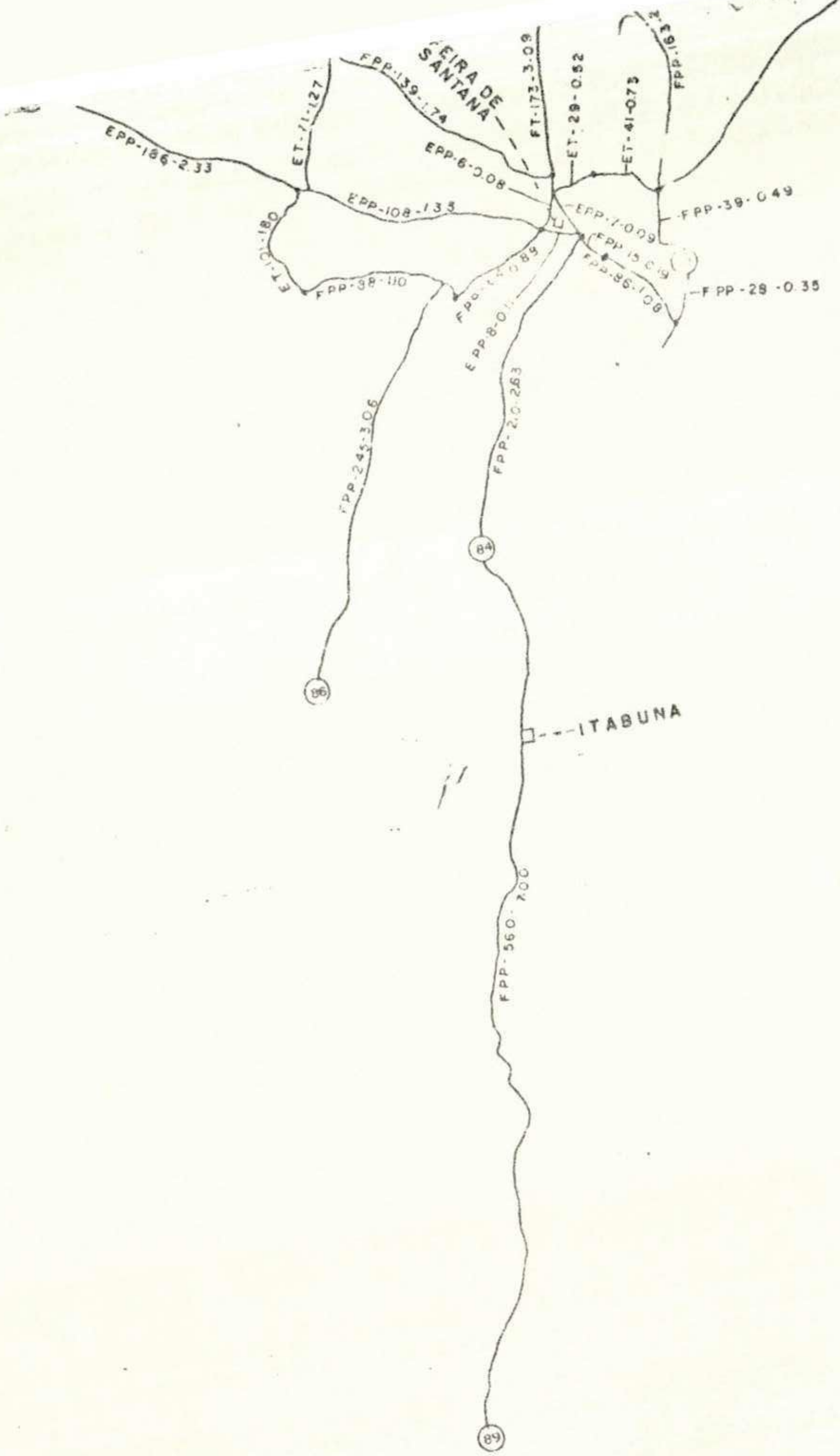
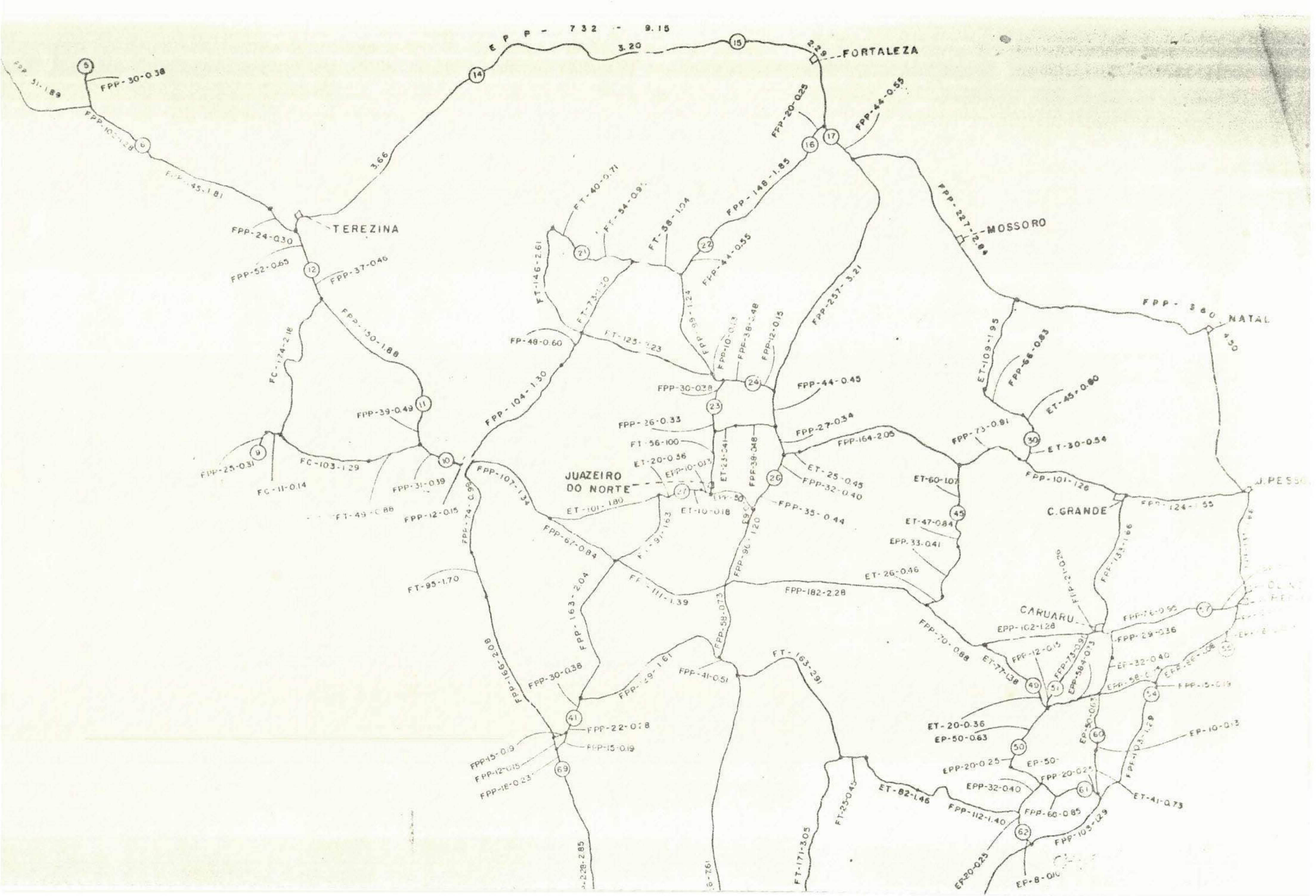


FIGURA IV.2 : RODOVIAS POSSIVEIS DE SEREM UTILIZADAS ENTRE AS ETEPAS 2 e 3 (ARRANJOS ÓTIMOS DE DESLOCAMENTO DE EQUIPAMENTO ENTRE DIFERENTES PÓLOS)

CONVENÇÕES:
FPP : ESTRADA PAVIMENTADA FEDERAL
FP : " EM PAVIMENTAÇÃO FEDERAL
FT : " DE TERRA FEDERAL
FC : " EM CONSTRUÇÃO FEDERAL
EPP : " PAVIMENTADA ESTADUAL
EP : " EM PAVIMENTAÇÃO ESTADUAL
ET : " DE TERRA ESTADUAL

FPP-63-0.79
TIPO DE ESTRADA
DISTANCIA EM KM
TEMPO DE VIAGEM EM HORA



FORTALEZA
TEREZINA
MOSSORO
NATAL
J. PESSO
C. GRANDE
CARUARU
JUAZEIRO
DO NORTE
E P P - 732 - 9.15
3.20
2.29
3.66
4.50
FPP-30-0.38
FPP-45-1.81
FPP-24-0.30
FPP-52-0.65
FPP-37-0.46
FPP-130-1.88
FC-174-2.18
FPP-39-0.49
FC-103-1.29
FPP-25-0.31
FC-11-0.14
FPP-31-0.39
FPP-12-0.15
FPP-20-0.25
FPP-44-0.55
FPP-148-1.85
FPP-227-12.84
FPP-257-3.21
FT-40-0.71
FT-58-1.04
FT-146-2.61
FP-48-0.60
FT-123-3.23
FPP-104-1.30
FPP-30-0.38
FPP-26-0.33
FT-56-100
ET-20-0.36
FPP-44-0.45
FPP-27-0.34
FPP-164-2.05
ET-25-0.45
FPP-32-0.40
FPP-35-0.44
FPP-107-1.34
ET-101-180
ET-10-0.18
FPP-67-0.84
FT-95-1.70
FPP-166-2.08
FPP-163-2.04
FPP-182-2.28
FPP-22-0.78
FPP-15-0.19
FPP-41-0.51
FT-163-2.91
FT-171-3.05
FT-25-0.45
ET-82-1.46
FPP-112-1.40
EP-20-0.25
ET-109-1.95
ET-45-0.80
FPP-73-0.81
ET-30-0.54
FPP-101-1.26
ET-60-1.07
ET-47-0.84
EPP-33-0.41
ET-26-0.46
FPP-124-1.55
FPP-133-1.66
EPP-102-1.28
FPP-70-0.88
ET-77-1.38
ET-20-0.36
EP-50-0.63
EPP-20-0.25
EPP-32-0.40
FPP-60-0.85
ET-41-0.73
EP-10-0.13
FPP-29-0.36
EP-8-0.10

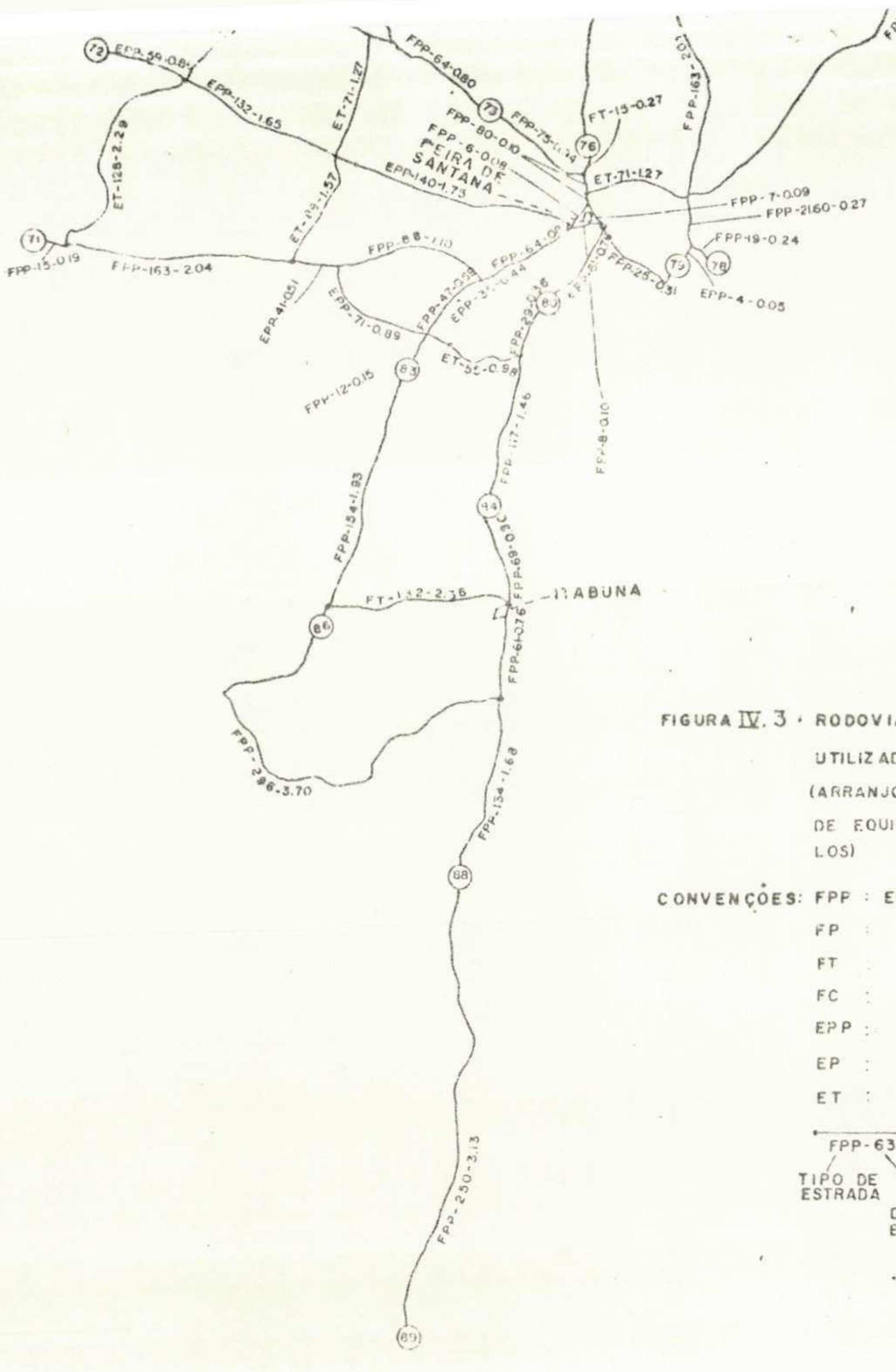


FIGURA IV. 3 · RODOVIAS POSSIVEIS DE SEREM UTILIZADAS ENTRE AS ETEPAS 3 e 4 (ARRANJOS ÓTIMOS DE DESLOCAMENTO DE EQUIPAMENTO ENTRE DIFERENTES PO LOS)

CONVENÇÕES:

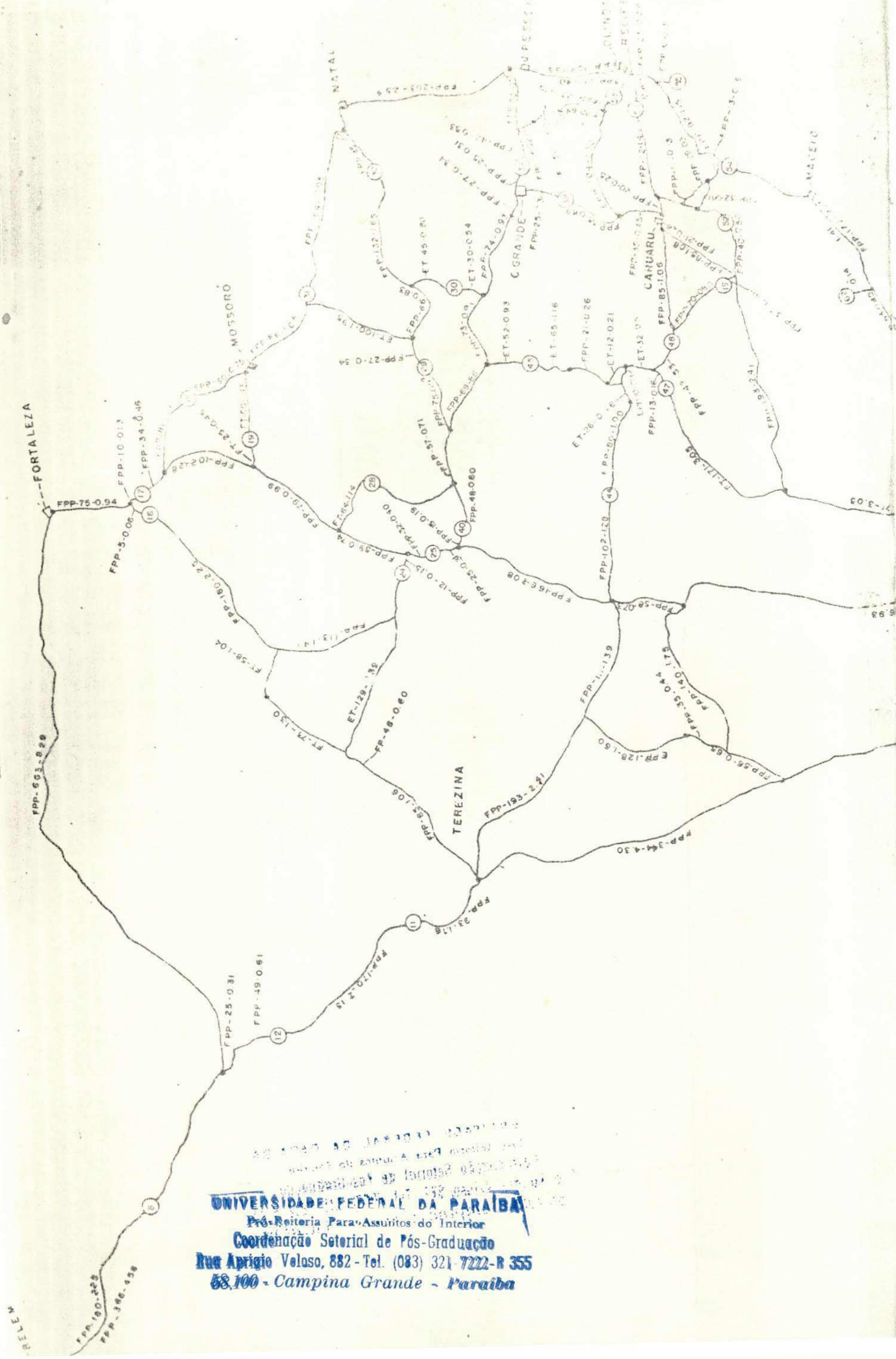
FPP	: ESTRADA	PAVIMENTADA FEDERAL
FP	: "	EM PAVIMENTAÇÃO FEDERAL
FT	: "	DE TERRA FEDERAL
FC	: "	EM CONSTRUÇÃO FEDERAL
EPP	: "	PAVIMENTADA ESTADUAL
EP	: "	EM PAVIMENTAÇÃO ESTADUAL
ET	: "	DE TERRA ESTADUAL

FPP-63-0.79

TIPO DE ESTRADA

DISTANCIA EM KM

TEMPO DE VIAGEM EM HORA


FORTALEZA
MOSSORÓ
NATAL
C. GRANDE
CARUARU
TEREZINA
MACEIO
FPP-75-0.94
FPP-663-829

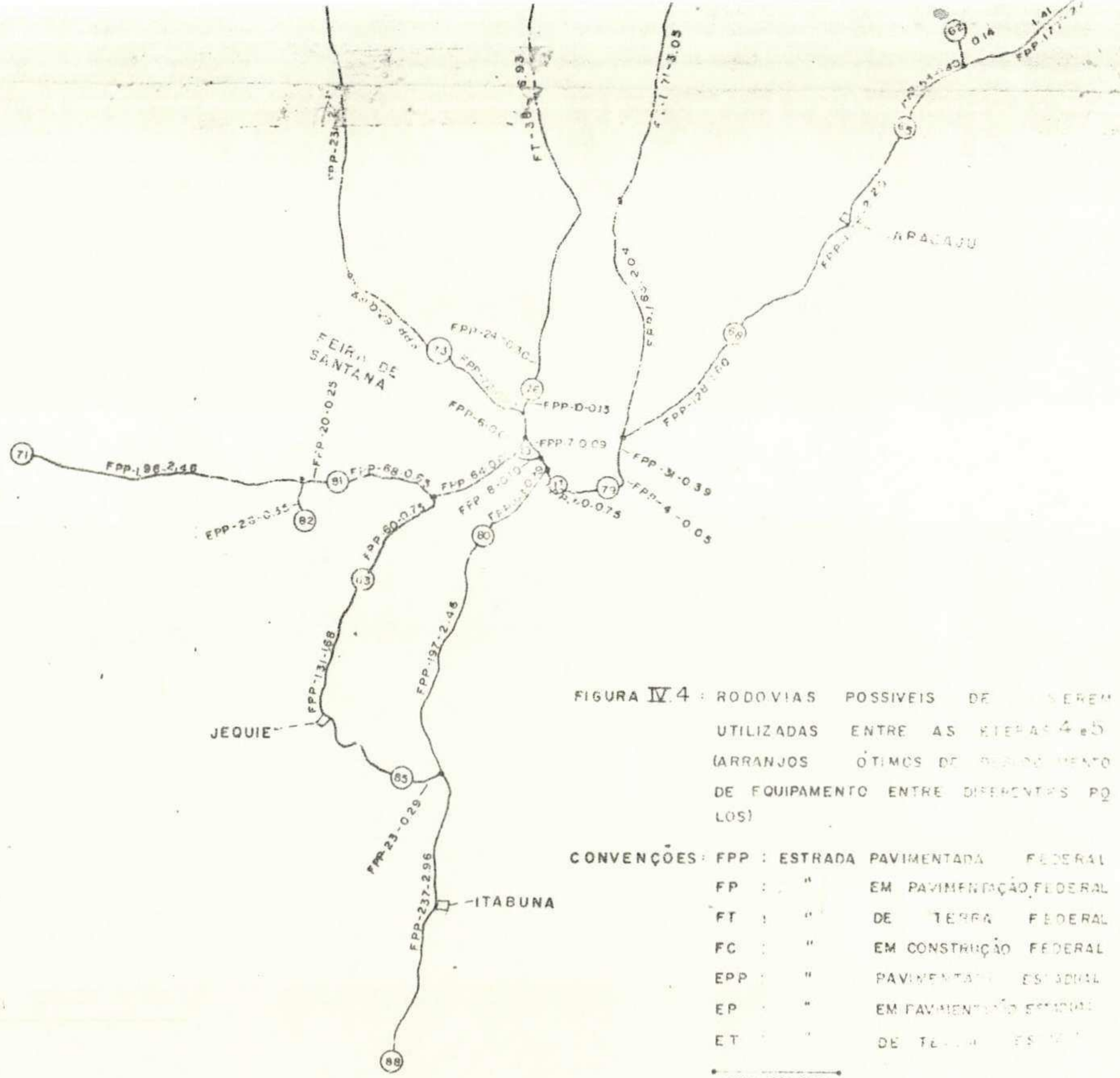
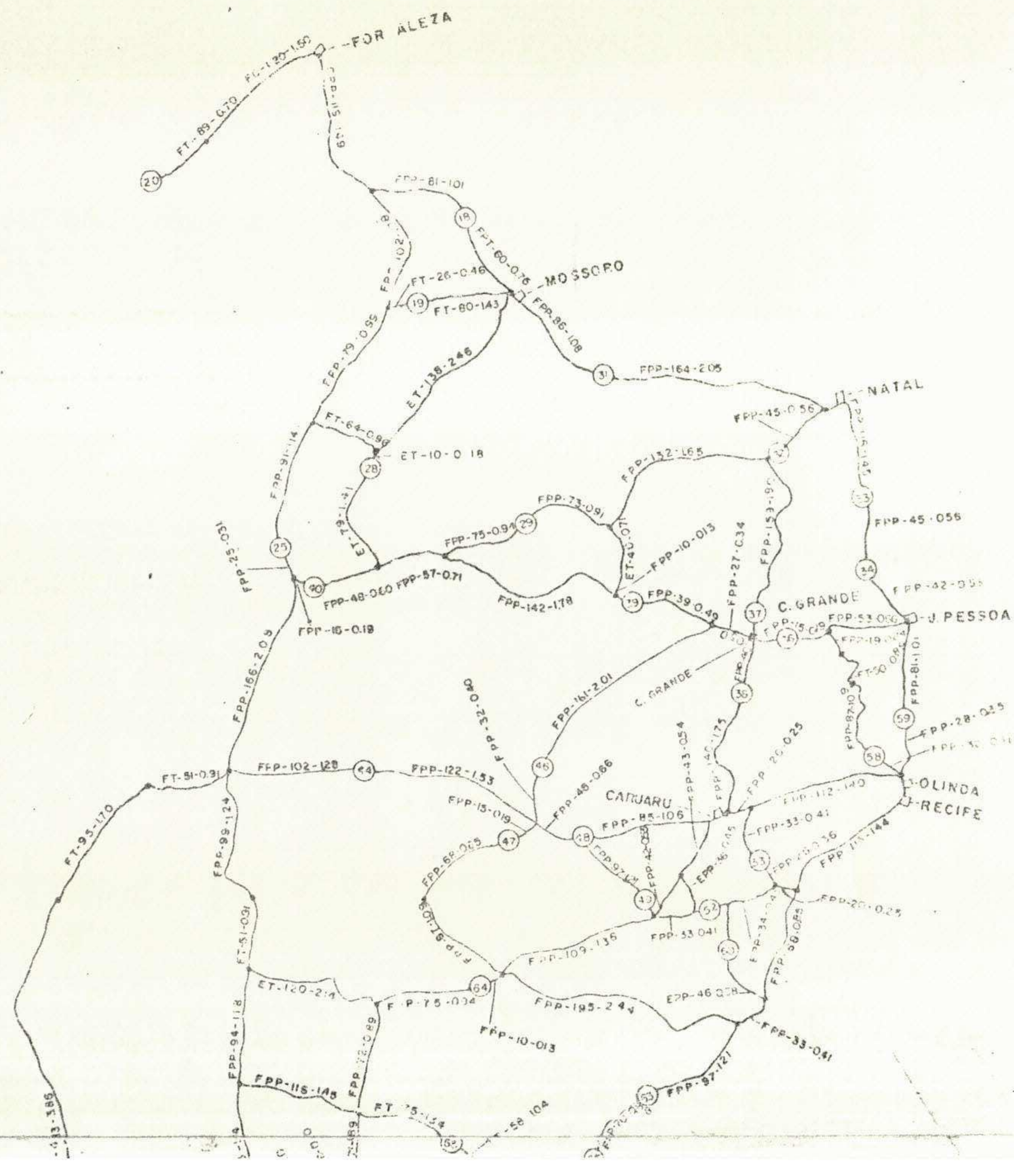


FIGURA IV.4 : RODOVIAS POSSIVEIS DE SEREM UTILIZADAS ENTRE AS ETEPAS 4 e 5 (ARRANJOS ÓTIMOS DE DESLOCAMENTO DE EQUIPAMENTO ENTRE DIFERENTES PÓLOS)

CONVENÇÕES: FPP : ESTRADA PAVIMENTADA FEDERAL
FP : " EM PAVIMENTAÇÃO FEDERAL
FT : " DE TERRA FEDERAL
FC : " EM CONSTRUÇÃO FEDERAL
EPP : " PAVIMENTADA ESTADUAL
EP : " EM PAVIMENTAÇÃO ESTADUAL
ET : " DE TERRA ESTADUAL

FPP- 63-0.79
TIPO DE ESTRADA
DISTÂNCIA EM KM
TEMPO DE VIAGEM EM HORA



FOR ALEZA
MOSSORO
NATAL
C. GRANDE
J. PESSOA
C. GRANDE
CARUARU
OLINDA
RECIFE
FT-89-0.70
FPP-115-149
FPP-81-101
FPT-60-0.76
FT-26-0.46
FT-80-143
FPP-86-108
FPP-164-2.05
FPP-79-0.55
ET-138-2.46
FT-64-0.96
ET-10-0.18
FPP-91-114
FPP-45-0.56
FPP-132-165
FPP-73-091
FPP-75-0.94
FPP-10-013
FPP-27-034
FPP-45-056
FPP-42-055
FPP-25-031
FPP-48-060
FPP-57-0.71
FPP-142-1.78
FPP-39-0.49
FPP-16-0.19
FPP-53-080
FPP-81-101
FPP-161-2.01
FPP-32-040
FPP-166-2.09
FPP-28-025
FPP-20-0.25
FPP-43-054
FT-51-0.31
FPP-102-128
FPP-122-1.53
FPP-46-066
FPP-15-019
FPP-85-106
FPP-112-140
FPP-33-041
FPP-68-085
FPP-99-124
FT-95-170
FPP-109-136
FPP-33-041
FPP-20-0.25
FT-51-031
ET-120-214
FPP-10-013
FPP-195-2.44
EPP-46-058
FPP-33-041
FPP-94-118
FPP-116-145
FPP-72-089
FPP-97-121

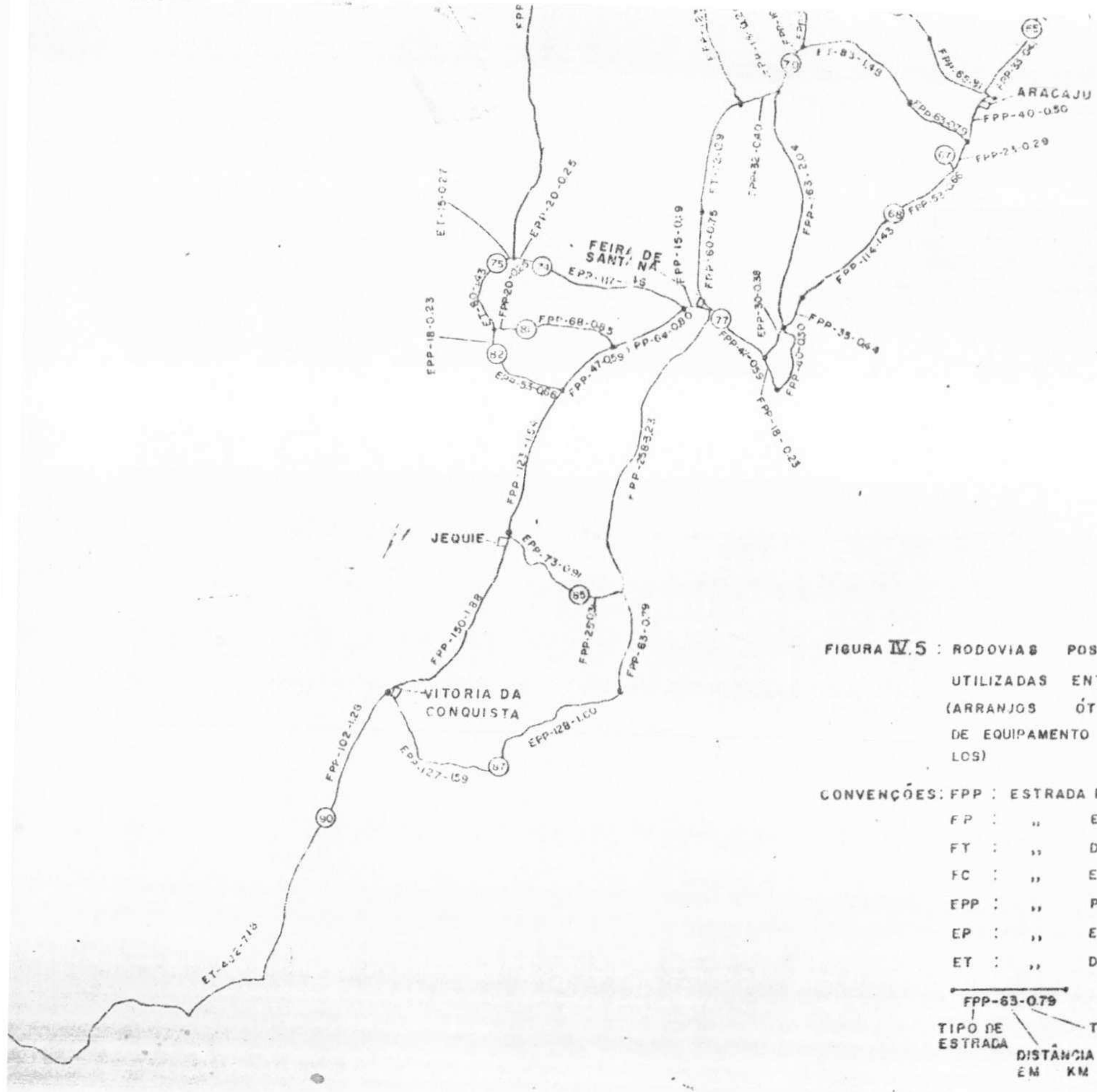


FIGURA IV.5 : RODOVIAS POSSIVEIS DE SEREM UTILIZADAS ENTRE AS ETAPAS 5 e 6 (ARRANJOS ÓTIMOS DE DESLOCAMENTO DE EQUIPAMENTO ENTRE DIFERENTES PÓLOS)

CONVENÇÕES:

FPP	:	ESTRADA PAVIMENTADA FEDERAL
FP	:	,, EM PAVIMENTAÇÃO FEDERAL
FT	:	,, DE TERRA FEDERAL
FC	:	,, EM CONSTRUÇÃO FEDERAL
EPP	:	,, PAVIMENTADA ESTADUAL
EP	:	,, EM PAVIMENTAÇÃO ESTADUAL
ET	:	,, DE TERRA ESTADUAL

FPP-63-0.79

TIPO DE ESTRADA — DISTÂNCIA EM KM — TEMPO DE VIAGEM EM HORA

3º PASSO - Para se converter as distâncias em tempo de viagem dividimos estas distâncias pelas velocidades permitidas em cada trecho. A velocidade permitida para estradas Pavimentadas (e em Pavimentação) é de 80 km/h e para estradas de terra de 56 km/h. Os tempos calculados são mostrados nos gráficos (IV.1) a (IV.5).

4º PASSO - Utilizando o algoritmo de FLOYD (Cap. V) obteve-se o tempo total minimizado entre todos os pares de vértices.

5º PASSO - O tempo total de viagem de cada origem a todos os destinos foram tirados dos resultados obtidos do Passo 4. Utilizou-se os resultados desta etapa como dados considerados na pesquisa. Tabelas (IV.2) a (IV.6).

IV.3 - CONSIDERAÇÕES GERAIS

Na fase de manipulação dos dados foram feitas as seguintes considerações:

1 - Devido à existência do grande número de vértices no grafo em questão, e, considerando-se a Capacidade limitada da memória do Computador, alguns deles foram suprimidos ao se entrar com os dados. O cancelamento deles foi feito de acordo com as seguintes condições:

(i) Se, para atingir um ou mais vértices existe apenas um caminho, este será desprezado. Estes vértices não considerados pelo Computador foram-no depois, manualmente quando necessário.

(ii) Muitos vértices que não influenciam na determinação do tempo total de viagem entre os Postos, foram ignorados.

2 - Outro aspecto levado em conta foi o de que a cidade de Campina Grande ficou considerada como centro de distribuição de equipamentos.

	04	15	42	43	56	36
67	3.31	10.56	11.30	11.55	17.80	18.62
13	6.75	5.20	10.87	10.21	16.46	15.45
C.G	18.58	10.83	7.25	6.29	3.46	1.08

TABELA IV.2 - TEMPO DA VIAGEM MINIMIZADA ENTRE OS POSTOS DA PRIMEIRA E SEGUNDA ETAPA

	09	10	05	06	21	22	26	27	57	51	61	50	62	41	69	78	84	86	89	72
04	6.73	9.66	1.54	2.82	10.33	12.71	14.66	13.22	19.98	19.19	20.93	20.01	21.72	14.85	15.12	21.42	22.43	23.65	29.43	21.57
[illegible]	[illegible]	4.51	6.33	7.65	9.12	9.59	9.81	8.37	15.13	14.34	16.08	15.16	16.87	10.00	10.27	16.57	17.58	18.80	24.58	16.72
42	7.90	4.83	12.95	11.67	6.97	5.35	1.69	2.01	5.95	5.16	6.90	5.98	7.23	2.44	3.37	9.67	10.68	11.90	17.68	9.82
43	7.80	4.73	12.85	11.57	6.61	4.99	1.33	1.65	5.85	5.06	6.80	5.88	7.59	2.80	3.73	10.03	11.04	12.26	18.04	10.18
56	13.84	10.77	18.89	17.61	12.44	10.82	7.63	7.95	0.67	2.23	2.46	2.89	3.63	8.84	9.77	10.91	11.92	13.14	18.92	12.96
35	13.97	10.90	19.02	17.74	10.29	8.67	6.21	7.65	2.50	3.83	4.78	4.65	5.95	8.97	9.90	13.23	14.24	15.46	21.24	15.28
C.3	12.90	8.62	17.34	16.66	9.21	7.59	5.13	6.57	2.78	2.75	3.73	3.57	4.90	7.89	8.82	12.18	13.19	14.41	20.19	14.23
01	12.16	14.71	12.87	11.59	13.06	13.52	19.71	18.27	25.03	24.24	25.98	25.06	26.77	19.90	20.17	26.47	27.48	28.70	34.48	26.62
02	12.14	14.59	11.59	10.31	11.78	12.24	20.99	19.55	26.31	25.52	27.26	26.34	28.05	21.18	21.45	27.75	28.76	29.98	35.76	27.30

TABELA IV.3 - TEMPO DA VIAGEM MINIMIZADO ENTRE OS POSTOS DA SEGUNDA E TERCEIRA ETAPA

	11	12	03	16	17	23	24	30	45	54	55	49	73	76	63	79	80	88	71
09	2.97	2.95	8.50	9.76	10.01	7.76	7.86	11.72	10.22	13.07	13.25	11.23	11.82	13.03	12.02	13.56	16.21	21.96	15.46
10	0.88	3.36	8.91	6.89	7.14	4.89	4.99	8.85	7.35	10.20	10.83	8.36	8.95	10.16	11.71	10.69	13.34	19.09	12.59
05	6.52	4.04	2.27	12.79	12.54	12.29	12.39	16.25	14.75	17.60	17.78	15.76	16.35	17.56	19.11	18.09	20.74	26.49	19.99
15	9.99	7.51	11.76	2.54	2.39	6.30	6.30	9.36	10.16	16.77	15.50	13.62	16.19	17.40	18.95	17.93	20.58	26.33	19.63
14	6.79	4.31	8.56	5.74	5.59	9.50	9.50	12.56	13.36	13.57	12.40	16.82	19.39	20.60	22.15	21.13	23.78	29.53	23.03
21	5.29	7.77	13.32	4.46	4.65	3.75	3.85	8.39	8.01	12.45	12.63	11.45	12.96	14.17	15.72	14.70	17.35	23.10	16.60
22	5.92	8.40	13.95	1.85	2.10	2.30	2.40	6.94	6.56	11.00	11.18	10.00	12.58	13.77	15.32	14.30	16.95	22.70	16.20
26	5.83	8.31	13.86	5.30	5.05	1.96	1.44	3.90	3.52	7.96	8.14	6.18	8.75	9.96	11.51	10.49	13.14	18.89	12.39
27	4.63	7.11	12.66	5.61	5.86	1.46	2.32	5.10	4.72	8.34	8.52	6.50	8.80	10.01	11.56	10.54	13.19	18.94	12.44
51	7.78	10.26	15.81	10.02	9.77	6.24	6.16	4.63	2.47	1.65	2.64	0.51	7.16	8.13	9.21	7.06	10.84	16.59	10.70
61	5.81	8.29	13.84	8.05	7.80	4.27	4.19	3.02	0.50	3.22	3.40	2.59	5.19	6.40	7.95	6.38	9.58	15.33	8.83
50	7.41	9.89	15.44	9.65	9.40	5.87	5.79	4.62	2.10	2.43	3.42	0.99	6.79	7.35	8.43	6.28	10.06	15.81	10.43
62	6.95	9.43	14.98	9.19	8.94	5.41	5.33	4.16	1.64	3.83	4.54	2.39	6.33	5.95	7.03	4.86	8.66	14.41	9.97
41	5.73	8.21	13.76	9.66	9.41	5.67	5.80	8.26	6.25	9.10	9.28	7.26	4.39	5.60	7.15	6.13	8.78	14.53	6.03
69	6.18	8.66	14.21	10.40	10.15	6.61	6.54	9.00	6.99	9.84	10.02	8.00	3.65	4.86	6.41	5.39	8.04	13.79	7.29
78	11.82	14.30	19.34	14.06	13.81	10.28	10.20	9.03	6.51	8.90	9.41	7.46	2.57	1.80	2.98	0.83	4.61	10.36	6.06
84	13.64	16.12	21.67	16.96	16.71	13.18	13.10	11.93	9.41	11.80	12.31	10.36	3.81	3.14	3.03	3.09	1.82	7.07	6.51
86	14.59	17.07	22.55	18.02	17.77	14.24	14.17	12.99	10.47	12.87	3.92	11.42	4.76	4.09	2.00	4.15	3.65	5.52	5.78
89	21.61	21.09	26.64	21.93	21.68	18.15	18.07	16.90	14.38	16.77	17.28	15.33	8.78	8.11	8.00	8.06	6.79	3.13	19.22
72	11.85	14.33	19.88	16.07	15.82	12.28	12.21	13.71	11.19	13.58	14.09	12.14	3.62	4.81	5.63	4.67	7.26	13.01	3.34
C.G	9.33	11.31	8.8	8.3	8.6	5.6	4.6	1.90	3.0	4.1	3.1	3.71	10.36	11.33	12.41	10.26	10.04	19.79	13.90

TABELA IV.4 - TEMPO DA VIAGEM MINIMIZADA ENTRE OS POSTOS DA TERCEIRA E QUARTA ETAPA

	46	47	31	32	18	19	25	40	36	38	58	44	48	28	29	77	65	68	85	81	82	52	53
11	8.17	8.10	10.78	12.05	9.28	8.27	6.48	6.98	10.83	10.84	11.83	6.45	8.47	7.96	9.23	10.39	13.85	13.18	13.51	11.93	12.53	10.31	10.30
12	10.30	10.23	12.91	14.18	11.41	10.40	8.61	9.11	12.96	12.96	13.96	8.61	10.60	10.09	11.36	12.52	15.98	15.31	15.64	14.06	14.66	12.44	12.43
[illegible]	16.50	15.73	17.21	19.68	15.38	11.90	14.11	14.61	18.46	18.46	19.46	14.11	16.10	15.59	16.66	18.02	21.48	20.81	21.14	19.56	20.16	17.94	17.93
[illegible]	13.25	13.48	14.96	17.43	13.13	9.65	11.86	12.36	16.21	16.21	17.21	11.86	13.85	13.34	14.61	15.77	19.23	18.56	18.89	17.31	17.91	15.69	15.68
16	9.36	6.32	3.46	6.07	1.63	2.36	4.03	4.53	9.16	9.16	9.81	7.70	9.69	4.03	5.75	14.48	14.29	16.49	17.60	16.02	1.62	11.40	10.81
17	9.17	9.13	3.27	5.88	1.44	2.17	3.84	4.34	8.97	8.97	9.62	7.51	9.50	3.84	5.56	14.29	14.10	16.30	17.41	15.83	16.43	11.21	10.62
23	6.80	6.76	5.77	7.04	5.09	3.26	1.47	1.97	6.63	6.63	8.91	5.14	7.13	2.95	4.22	11.92	12.03	14.23	15.04	13.46	14.06	8.87	8.28
24	5.88	5.84	4.85	6.12	4.17	2.34	0.56	1.05	5.71	5.72	7.99	4.22	6.21	2.03	3.30	11.00	11.11	13.31	14.12	12.54	13.14	7.95	7.36
30	3.48	4.31	3.58	2.45	5.41	6.09	4.12	3.62	2.12	2.12	4.40	5.17	4.58	4.45	1.97	12.51	7.52	9.72	15.94	14.36	14.96	4.36	3.77
45	1.73	2.36	5.02	4.83	6.85	6.19	3.60	3.10	3.42	3.42	5.70	3.02	2.73	3.93	2.73	11.69	8.11	10.31	15.12	13.54	14.14	4.57	4.56
57	3.77	3.26	6.34	4.86	8.18	8.86	7.55	7.05	2.95	2.39	0.80	4.56	2.57	7.88	6.48	10.00	5.01	7.48	13.43	11.85	12.45	1.85	1.26
54	3.74	3.23	7.58	6.09	9.41	10.09	7.52	7.02	2.98	2.36	2.03	4.53	2.54	7.85	6.45	8.24	3.25	5.45	11.67	10.09	10.69	1.34	0.71
55	4.59	4.08	6.37	4.88	8.20	8.88	8.13	7.63	2.97	3.10	0.82	5.38	3.33	8.46	7.06	9.71	4.72	6.92	13.14	11.56	12.16	2.43	1.80
42	2.08	1.57	8.62	7.49	10.45	9.13	6.54	6.42	3.54	2.92	3.61	2.87	0.88	7.54	6.34	9.15	4.50	6.70	12.56	11.00	11.60	0.96	1.61
62	5.92	5.41	10.44	9.21	12.27	12.49	9.90	9.40	5.36	4.74	5.15	6.71	4.72	10.23	8.83	5.93	3.14	3.14	9.36	7.78	8.38	2.88	3.35
73	9.85	9.78	13.69	14.22	13.69	11.86	9.27	9.15	12.19	11.57	11.45	8.16	10.15	11.18	11.40	1.18	6.17	3.97	4.30	2.72	3.32	9.71	10.13
76	10.57	9.74	14.72	14.74	14.72	12.89	10.30	10.18	11.42	10.50	10.68	9.19	10.43	12.21	12.43	0.41	5.40	3.20	3.53	1.95	2.55	8.94	9.36
83	11.91	11.08	16.31	16.08	16.31	14.48	11.89	11.77	12.76	12.14	12.02	10.78	11.77	13.60	14.02	1.75	6.74	4.54	1.68	1.60	2.20	10.28	10.70
79	9.41	8.56	15.07	13.56	15.62	13.79	11.20	11.08	10.26	9.64	9.52	10.09	9.27	13.11	13.33	0.75	4.24	2.04	4.18	2.60	3.20	7.78	8.20
80	10.95	10.12	15.66	15.12	15.66	13.83	11.24	11.12	11.80	11.18	11.06	10.13	10.81	13.15	13.37	0.79	5.78	3.58	2.75	2.64	3.24	9.32	9.74
69	16.37	15.54	21.08	20.54	21.08	19.25	16.66	16.54	17.22	16.60	16.48	15.55	16.23	18.57	18.79	6.21	11.20	9.00	3.25	6.53	7.13	11.78	12.20
71	14.71	13.88	19.11	18.88	19.11	17.28	14.69	14.57	15.56	14.94	14.82	13.58	14.57	16.60	16.82	4.55	9.54	7.34	5.98	2.70	2.80	13.08	13.50
[illegible]	[illegible]	3.15	5.33	4.26	7.22	7.90	4.55	4.35	0.31	0.31	2.59	4.04	2.77	5.18	3.78	12.51	5.71	7.91	14.13	12.55	13.15	2.55	1.96

TABELA IV.5 - TEMPO DA VIAGEM MINIMIZADA ENTRE OS POSTOS DA QUARTA E QUINTA ETAPA

	50	20	37	39	64	66	59	63	67	33	34	91	92	87	90	70	74	75
[illegible]	[illegible]	10.70	2.54	[illegible]	2.66	5.93	4.31	5.61	7.66	4.94	4.38	22.04	22.08	13.64	14.50	5.58	9.87	10.39
[illegible]	[illegible]	[illegible]	3.13	[illegible]	2.07	5.24	4.10	5.40	7.09	5.53	4.97	21.45	21.49	13.25	13.91	4.99	9.28	9.80
31	8.63	6.53	4.60	5.10	9.80	12.97	5.60	9.86	11.94	3.50	4.06	28.22	26.26	20.02	20.68	12.72	15.18	15.20
32	6.33	9.14	1.99	2.49	7.19	10.36	4.11	8.37	10.45	2.01	2.57	26.57	26.61	18.37	19.03	10.16	14.45	14.97
18	10.46	4.70	6.43	6.93	11.63	12.93	7.43	11.69	13.77	5.33	5.69	27.54	27.58	19.34	20.00	12.28	14.50	14.52
19	10.80	5.43	7.11	6.31	10.05	11.10	8.11	12.37	13.21	6.01	6.57	25.71	25.75	17.51	18.17	10.45	12.67	12.69
25	8.21	7.10	4.74	3.72	7.46	8.51	7.06	10.41	10.62	7.14	6.58	23.12	23.16	14.92	15.58	7.86	10.08	10.10
40	8.09	7.60	4.24	3.22	7.34	8.39	6.56	10.01	10.50	6.64	6.11	23.00	23.04	14.80	15.46	7.74	9.96	9.98
35	4.49	11.68	0.53	1.17	5.35	6.42	2.17	6.43	8.51	2.25	1.69	24.73	24.77	16.53	17.19	8.32	12.61	13.13
36	3.65	11.52	0.69	1.33	4.56	7.73	3.01	5.46	7.54	3.09	2.53	23.94	23.98	15.74	16.40	7.48	11.77	12.29
[illegible]	[illegible]	[illegible]	3.25	3.89	4.78	7.45	0.75	4.31	6.39	2.65	2.29	23.77	23.81	15.61	16.23	7.70	11.50	12.02
44	4.54	10.77	4.47	4.43	3.97	6.96	5.44	6.74	8.81	6.67	6.31	22.01	22.05	13.81	14.47	6.71	8.97	8.99
48	2.41	12.90	3.54	3.50	2.83	6.00	3.31	4.61	6.69	5.41	4.85	22.24	22.28	14.04	14.70	5.78	10.07	10.59
28	9.01	6.92	5.05	4.03	9.19	10.45	7.37	10.62	12.07	7.22	7.07	25.01	25.05	16.81	17.47	9.71	11.97	11.99
29	6.73	9.85	2.77	1.75	6.91	10.08	5.09	8.54	10.62	4.57	4.61	25.25	25.29	17.05	17.71	9.63	12.21	12.23
77	7.78	16.50	10.86	10.82	5.66	5.17	9.84	5.58	3.50	11.94	11.38	13.92	13.96	5.62	6.38	2.74	1.75	2.27
65	3.08	17.27	7.03	7.67	4.66	2.26	5.14	0.88	1.20	7.24	6.68	18.58	18.62	10.28	11.04	3.38	6.31	6.83
68	4.94	18.81	8.89	9.53	5.94	3.30	7.00	2.74	0.66	9.10	8.54	16.72	16.76	8.42	9.18	3.42	4.59	5.11
85	11.65	20.17	14.53	14.63	9.53	9.04	13.71	9.45	7.37	15.81	15.25	11.61	11.65	2.93	4.07	6.61	5.29	4.77
61	9.72	18.24	12.60	12.56	7.40	6.91	11.78	7.52	5.44	13.88	13.32	13.38	13.42	6.15	5.84	4.48	2.20	1.68
62	10.13	18.64	13.00	12.96	7.80	7.31	12.18	7.92	5.84	14.28	13.72	12.90	12.94	5.67	5.36	4.88	2.18	1.66
52	0.59	14.75	3.78	4.42	2.00	5.17	2.73	2.76	4.84	4.83	4.27	21.38	21.42	13.18	13.84	4.92	9.21	9.73
53	1.32	14.23	3.10	3.74	2.89	6.19	2.16	3.05	5.13	4.26	3.70	22.40	22.44	14.20	14.84	5.94	10.23	10.75

TABELA IV.6 - TEMPO DA VIAGEM MINIMIZADA ENTRE OS POSTOS DA QUINTA E SEXTA ETAPA

n.6 5.6.

CAPÍTULO V

OTIMIZAÇÃO EM REDES

Neste capítulo serão apresentados os algoritmos utilizados para resolver o problema. Inicialmente, apresenta-se o algoritmo de MOORE (13), BELLMAN (1) e FORD (8) para encontrar o menor caminho entre dois vértices especiais - s e t - e o algoritmo de FLOYD (7) para determinar os menores caminhos entre todos os pares de vértices. Em seguida, apresentam-se os algoritmos para detetar circuito negativo existente no grafo e, finalmente, os algoritmos para se determinar um fluxo que obtenha um custo mínimo.

V.1 - MENOR CAMINHO

Em um grafo de arcos ponderados G (X, Γ) , para o qual os custos são representados pela matriz $C = (C_{ij})$ o problema é encontrar o menor caminho de um vértice $s \in X$ a um vértice especial $t \in X$. Considerou-se que $t \in R(s)$ onde $R(s)$ é o conjunto de vértices que pode ser atingido do vértice s. Os elementos C_{ij} da matriz de custo C podem ser positivos, negativos ou nulos, estabelecendo-se que não existe nenhum circuito em G, cujo custo total seja negativo. Se existir um circuito $\emptyset$ e x_i sendo um dos seus vértices, então para se chegar a t, do vértice inicial s, atravessa-se ao longo do circuito $\emptyset$ muitas vezes, e, finalmente, chega-se a t com um custo razoavelmente pequeno $(-\infty)$ e o melhor caminho não é unicamente definido.

V.1.1 - MENOR CAMINHO DE s PARA OUTROS VÉRTICES

Considera-se que os elementos da matriz de custo podem ser positivos, negativos ou nulos. O custo negativo associado aos arcos, representa os arcos vantajosos. Para calcular os menores caminhos de um vértice para todos os outros vértices, usa-se um método iterativo, baseado no endereçamento de vértices, onde, no fim da iteração κ, os endereços representam os valores dos menores caminhos que contém $(\kappa + 1)$ arcos ou menos. O método originalmente foi proposto em 1950 por FORD (8), MOORE (13) e BELLMAN (1).

Descrição - Supõe-se $L^{\kappa}(x_i)$ o endereço do vértice x_i após a interação $(\kappa + 1)$.

Inicialização:

Passo 1 - Faz $S = \Gamma(s)$, $\kappa = 1$, $L^1(s) = 0$, $L^1(x_i) = c(s, x_i)$ para tudo $x_i \in \Gamma(s)$ e $L^1(x_i) = \infty$ para todos os outros vértices.

Atualizando os endereços:

Passo 2 - Para todos os vértices $x_i \in \Gamma(S)$, $(x_i \neq s)$ atualiza-se o endereço, de acordo com a seguinte expressão:

$$L^{\kappa+1}(x_i) = \min \{L^{\kappa}(x_i), \min_{x_j \in T_i} \left[L^{\kappa}(x_j) + C(x_j, x_i) \right] \} \quad \text{(Equ. V.1.1)}$$

Onde $T_i = \Gamma^{-1}(x_i) \cap S$. (Portanto os elementos do conjunto S são os vértices de cardinalidade κ no menor caminho). O conjunto T_i contém os vértices, no qual o menor caminho de s é de cardinalidade κ. (i.e. os elementos do conjunto S). Note-se que se $x_i \in \Gamma(S)$ o menor caminho de s para x_i, possivelmente não pode ser de cardinalidade $\kappa + 1$ e não é necessário mudar os endereços de x_i, para os vértices $x_i \notin \Gamma(S)$, faz-se $L^{\kappa+1}(X_i) = L^{\kappa}(x_i)$.

Teste de Finalização:

Passo 3 - (a) se $\kappa \leq n-1$ e $L^{\kappa+1}(x_i) = L^{\kappa}(x_i)$ para todos os x_i, então, a solução ótima está atingida e os endereços são os custos dos menores caminhos. Pare.

(b) Se $\kappa < n-1$ mas $L^{\kappa+1}(x_i) \neq L^{\kappa}(x_i)$ para algum x_i, vã para o Passo 4.

(c) Se $\kappa = n-1$ e $L^{\kappa+1}(x_i) \neq L^{\kappa}(x_i)$ para algum x_i, então um circuito com custo negativo, existe no grafo e não existe solução ōtima. Pare.

Passo 4 - Atualize o conjunto S da seguinte maneira:

$$S = \{x_i / L^{\kappa+1}(x_i) \neq L^{\kappa}(x_i)\}$$

(O conjunto S contēm aqueles vērtices cujos menores caminhos de s são de cardinalidade $\kappa+1$)

Passo 5 - Faz-se $\kappa = \kappa+1$ e volta-se para o Passo 2.

Desde que os custos dos menores caminhos de s para todos os vērtices tenham sido obtidos, aplicando-se a equ. (V.1.2) por diversas vezes, pode-se achar os menores caminhos.

$$L(x_i') + C(x_i', x_i) = L(x_i) \qquad \text{(Equ. V.1.2)}$$

(x_i' ē o vertice precedente ao vertice x_i no menor caminho). Alternativamente, os caminhos podem ser obtidos imediatamente se, se armazenar outro endereço $\theta^{\kappa}(x_i)$ para cada vērtice durante a computação, onde $\theta^{\kappa}(x_i)$ ē o vērtice procedente do vērtice x_i no menor caminho de s para x_i na iteração κ. Pode-se começar com $\theta^1(x_i) = s \ \forall \ x_i \ \varepsilon \Gamma(s)$ e $\theta^1(x_i) = 0$ (arbitrārio) para todos os outros vértices x_i. Os endereços $\theta^{K}(x_i)$ podem ser atualizados depois da equação (V.1.1). Se o primeiro termo for menor, então, faz-se $\theta^{K+1}(x_i) = \theta^{K}(x_i)$ ou faz-se $\theta^{K+1}(x_i) = x_j$ se o segundo termo no colchete for menor. No fim do algoritmo, pode se obter o menor caminho de s para x_i na ordem reversa como s, ... $\theta^3(x_i), \theta^2(x_i), \theta(x_i)$: Onde $\theta^2(x_i)$ é escrito para $\theta(\theta(x_i))$.

A prova que o resultado obtido é realmente ótimo é baseado no princípio de otimização de programação dinâmica e no fato de que se não exisitir um caminho ótimo de K arcos, não pode exisitir um caminho ótimo de $K+1$ arcos. No caso de um grafo completamente interligado de n vértices, o algoritmo exige uma ordem de n^3 operações (adições e comparações). Vários melhoramentos a esse algoritmo foram feitos por YEN (15), que reduz o esforço computacional por um fator de quatro, mesmo que a dependência em números de vértices no grafo seja cubica.

Ilustração (4) - considere o grafo da Figura (V.1) na qual os ramos serão considerados como dois arcos com custos iguais em qualquer uma das suas direções. Os custos são mostrados junto com os arcos, e contém números positivos e negativos. Deseja-se determinar o menor caminho de x_1 para todos os outros vértices (supondo que o grafo não contém nenhum circuito de custo negativo) ou indicando-se se existirem.

O algoritmo procede da seguinte maneira:

Inicialização:

Passo 1 - $s = x_1$, $S = \{x_2, x_5\}$. $L^1(x_2) = -3$, $L^1(x_5) = 2$, $L^1(x_i) = \infty$ para todos os outros vértices $x_i \notin S$. Faz-se $\kappa=1$.

Primeira Iteração

Passo 2 - $\Gamma(S) = \{x_2, x_3, x_4, x_5, x_6\}$. Portanto.

Para x_2 : $T_2 = \{x_1, x_5\} \cap \{x_2, x_5\} = \{x_5\}$ e da equ.(V.1.1):

$$L^2(x_2) = \min\left[-3, \{L^1(x_5) + C(X_5, x_2)\}\right]$$

$$= \min\left[-3, (2 + 1 + 2)\right]$$

$$= -3$$

Para x_3 : $T_3 = \{x_2, x_7, x_8, x_4\} \cap \{x_2, x_5\} = \{x_2\}$

$$L^2(x_3) = \min\left[\infty, \underset{x_j = x_2}{(-3 - 5)}\right] = -8$$

Para x_4 : $T_4 = \{x_2, x_3, x_5, x_7, x_9\} \cap \{x_2, x_5\} = \{x_2, x_5\}$

$$L^2(x_4) = \min\left[\infty, \min\{\underset{x_j = x_2}{(-3 + 15)}, \underset{x_j = x_5}{(2-7)}\}\right] = -5$$

Para x_5 : $T_5 = \{x_1, x_2, x_6\} \cap \{x_2, x_5\} = \{x_2\}$

$$L^2(x_5) = \min\left[2, \underset{x_j = x_2}{(-3 + 12)}\right]$$

Para x_6 : $T_6 = \{x_4, x_5, x_7, x_9\} \cap \{x_2, x_5\} = \{x_5\}$

$$L^2(x_6) = \min \left[\infty, \underset{x_j = x_5}{(2+20)} \right] = 22$$

Agora os endereços são $\left[0, -3, -8, -5, 2, 22, \infty, \infty, \infty \right]$ para $x_i = x_1, x_2, \ldots, x_9$ respectivamente.

Passo 3 (b) - vá para o passo 4

Passo 4 - $S = \{x_3, x_4, x_6\}$

Passo 5 - $\kappa = 2$, vá para o Passo 2

Segunda Iteração

Passo 2 - $\gamma(S) = \{x_1, x_3, x_4, x_5, x_6, x_7, x_8, x_9\}$;

Para x_3 : $T_3 = \{x_2, x_4, x_7, x_8\} \cap \{x_3, x_4, x_6\} = \{x_4\}$

$$L^3(x_3) = \min \left[-8, \underset{x_j = x_4}{(-5 + 8)} \right] = -8$$

Para x_4 : $T_4 = \{x_2, x_3, x_5, x_7, x_9\} \cap \{x_3, x_4, x_6\} = \{x_3\}$

$$L^3(x_4) = \min \left[-5, \underset{x_j = x_3}{(-8 + 8)} \right] = -5$$

Para x_5 : $T_5 = \{x_1, x_2, x_6\} \cap \{x_3, x_4, x_6\} = \{x_6\}$

$$L^3(x_5) = \min \left[2, \underset{x_j = x_6}{(22 + 20)} \right] = 2$$

Para x_6 : $T_6 = \{x_4, x_5, x_7, x_9\} \cap \{x_3, x_4, x_6\} = \{x_4\}$

$$L^3 (x_6) = \min \Big[22, \underset{x_j = x_4}{(-5 + 18)} \Big] = 13$$

Para x_7 : $T_7 = \{x_4, x_6, x_8\} \cap \{x_3, x_4, x_6\} = \{x_4, x_6\}$

$$L^3 (x_7) \min \Big[\infty, \min \{ \underset{x_j = x_4}{(-5 + 4)}, \underset{x_j = x_6}{(22 + 9)} \} \Big] = -1$$

Para x_8 : $T_8 = \{x_3, x_7\} \cap \{x_3, x_4, x_6\} = \{x_3\}$

$$L^3 (x_8) = \min \Big[\infty, \underset{x_j = x_3}{(-8 + 24)} \Big] = 16$$

Para x_9 : $T_9 = \{x_4, x_8\} \cap \{x_3, x_4, x_6\} = \{x_4\}$

$$L^6 (x_9) = \min \Big[\infty, \underset{x_j = x_4}{(-5 + 11)} \Big] = 6$$

Os endereços $L^3 (x_i)$ são $\Big[0, -3, -8, -5, 2, 13, -1, 16, 6 \Big]$

para x_1, x_2 ..., x_9, respectivamente.

Passo 3 (b) - vá para o passo 4

Passo 4 - $S = \{x_6, x_7, x_8, x_9\}$

Passo 5 - $\kappa = 3$, vá para o passo 2

E T C.

Continuando, dessa maneira, obtém-se os seguintes resultados:

Terceira Iteração

Passo 2 - $\Gamma(S) = \{x_3, x_4, x_5, x_6, x_7, x_8, x_9\}$

$T_3 = \{x_7, x_8\}$, $L^4 (x_3) = -11$

$T_4 = \{x_7, x_9\}$, $L^4 (x_4) = -5$

$T_5 = \{x_6\}$, $L^4 (x_5) = 2$

$T_6 = \{x_7, x_9\}$, $L^3 (x_6) = -7$

$T_7 = \{x_6, x_8\}$, $L^4 (x_7) = -1$

$T_8 = \{x_7\}$, $L^4 (x_8) = 15$

$T_9 = \{x_8\}$, $L^4 (x_9) = 6$

Portanto, o vetor dos endereços $L^4 (x_i)$ é:

$$\left[0, -3, -11, -5, -7, 2, -1, 15, 6\right].$$

Passo 4 - $S = \{x_3, x_6, x_8\}$

Quarta Iteração

Passo 2 - $\Gamma(S) = \{x_3, x_4, x_5, x_7, x_8, x_9\}$

$T_3 = \{x_8\}$, $L^5 (x_3) = -11$

$T_4 = \{x_3\}$, $L^5 (x_4) = -5$

$T_5 = \{x_6\}$, $L^5 (x_5) = 2$

$T_7 = \{x_6, x_8\}$, $L^5 (x_7) = -1$

$T_8 = \{x_3\}$, $L^5 (x_8) = 13$

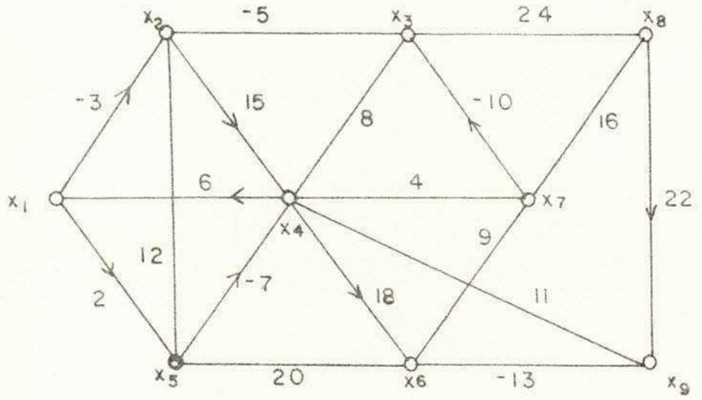


FIGURA V.1 — GRAFO PARA ILUSTRAÇÃO I

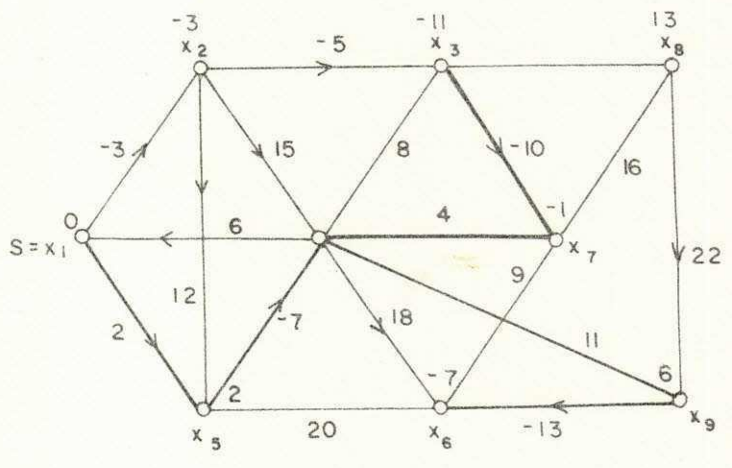


FIGURA V.2 — OS ENDEREÇOS FINAIS DOS VÉRTICES

$T_9 = \{x_8\}$, $L^5 (x_9) = 6$

Então o valor dos endereços é: $\left[0,-3,-11,-5,2,-7,-1,13,6\right]$

Passo 4 - $S = \{x_8\}$

Quinta Iteração

Passo 2 - $\Gamma(S) = \{x_3, x_7, x_9\}$

$T_3 = \{x_8\}$, $L^6 (x_3) = -11$

$T_7 = \{x_8\}$, $L^6 (x_7) = -1$

$T_9 = \{x_9\}$, $L^6 (x_9) = 6$

Passo 3 (a) - Pare.

O valor dos $L^6 (x_i)$ é o mesmo que $L^5 (x_i)$ e, por isso, esses são os custos do menor caminho (Fig. V.2).

V.1.2 - MENORES CAMINHOS ENTRE TODOS OS PARES DE VÉRTICES (ALGORITMO DE FLOYD)

Supõe-se que a matriz do custo tinha sido inicializado de maneira que $C_{ii} = 0$ para $i = 1, 2, ..., n$, e $C_{ij} = \infty$, quando o arco (x_i, x_j) não esteja no grafo.

Inicialização

Passo 1 - faz-se $\kappa = 0$

Iteração 1

Passo 2 - $\kappa = \kappa + 1$

Passo 3 - Para todo $i \neq \kappa$ quando $C_{i\kappa} \neq \infty$ e para todo $j \neq \kappa$, quando $C_{\kappa j} \neq \infty$, efetua-se a seguinte operação

$$C_{ij} = \min \left[C_{ij}, (C_{i\kappa} + C_{\kappa j}) \right] \qquad \text{(Equ. V.1.3)}$$

Teste de Terminação

Passo 4 (a) Se qualquer $C_{ii} < 0$, isto significa que um circuito com custo negativo, contendo o vértice x_i existe no grafo. Portanto não se pode obter uma solução ótima. Pare.

(b) Se todo $C_{ii} \geq 0$ e $\kappa = n$, a solução tinha sido atingida, e $\left[C_{ij} \right]$ dará o tamanho de todos os menores caminhos. Pare.

(c) Se todo $C_{ii} \geq 0$, mas $\kappa < n$ volte para o passo 2 e continui.

Os vértices dos menores itinerários podem ser obtidos dos tamanhos em menores caminhos, utilizando-se uma relação similar à equação (V.1.2).

Esse último método é parecido com o da Secção (V.1.1) e, geralmente, é utilizado, quando se deseja determinar a seqüência dos vértices no circuito negativo existente no grafo. A técnica implica em armazenar uma outra matriz (n x n) a matriz $\theta = \left[\theta_{ij} \right]$. A entrada θ_{ij} significa que

θ_{ij} é o vértice antecedente do vértice x_j no menor caminho de x_i para x_j. Inicializa-se a matriz de tal modo que $\theta_{ij} = x_i$ para todo x_i e x_j.

Acompanhamento de equação (V.1.2), no Passo 3 pode-se atualizar a matriz θ como se segue:

$$\theta_{ij} = \begin{cases} \theta_{\kappa j}, \text{ se } (C_{i\kappa} + C_{\kappa j}) < C_{ij} \\ \text{Sem mudar, se } C_{ij} \geq (C_{i\kappa} + C_{\kappa j}) \end{cases}$$

(no colchete da Equ. V.1.3)

Finalmente, pode-se obter os menores caminhos da matriz final de θ. Entretanto se se desejar o menor caminho entre qualquer dos dois vértices x_i e x_j e, esse caminho é dado pela seqüência dos vértices:

$$x_i, x_v, \ldots x_\gamma, x_\beta, x_\alpha, x_j$$

Onde $x_\alpha = \theta_{ij}$, $x_\beta = \theta_{ix}$, $x_\gamma = \theta_{i\beta}$ etc. até que $x_i = \theta_{iv}$

Inicialmente, todos os elementos da diagonal principal na matriz do custo (C_{ii}) são iguais a (∞). Os valores finais de C_{ii} serão o custo do menor circuito através do vértice x_i. Facilmente, pode-se ver que a condição de matriz θ na iteração em que C_{ii} tornou-se negativa identifica o circuito negativo, correspondente àquele vértice (x_i). A ilustração deste algoritmo virá no item (V.3.2).

V.2 - MÉTODO DA PESQUISA DIRETA PARA DETETAR O CIRCUITO NEGATIVO DA REDE (6)

O propósito dessa Secção é introduzir um método para localizar os circuitos negativos no grafo. Este método é baseado em propriedade de somatórios parciais negativos numa seqüencia finita.

V.2.1 - DESCRIÇÃO DO ALGORITMO

Supõe $C' = (C'_{ij})$ seja a matriz do custo para o grafo direcionado G (X, A)

Esse procedimento de pesquisa começa de um vértice, e constrói uma progressão de arcos com o custo negativo.

Se um somatório de custos não pode ser mantido, esse vértice é abandonado e da mesma forma outro vértice é considerado. Esse procedimento é repetido até que um circuito negativo seja alocado. Se um circuito negativo não existir no grafo, ainda todos os vértices são tratados.

Faz-se, Open, Cycle e Value vetores de n posições.

Passo 1 - i = 1

Passo 2 - Cycle (j) = value (j) = 0 (j = 1, 2, ..., n)

Open (i) = 1

Passo 3 - Open (i) = value (l) = 0

Cycle (l) = i

l = s = 1,

u = i

Passo 4 - Pesquisar na linha u da coluna s até a coluna n, e achar o menor índice V de maneira que:

Val ≡ {Open (v). [value (l) + C(u, v)]} < 0

Se, na pesquisa houver sucesso, vá para (7)

Passo 5 - Adicionar um novo ponto no circuito, faz-se

Open (u) = 0

l = l + 1

Cycle (l) = μ, value (l) = val

Passo 6 - Verificar se o circuito pode ser completo. Faz-se

ω = Cycle (l)

Se Value (l) + C (u, ω) < 0. Pare.

O circuito é completo.

Se não se faz s = l e volta para o Passo (4)

Passo 7 - O circuito não pode ser extendido, "back truck"

(a) Se l = 1, vá para o Passo (8), se não se deve fazer s = Cycle (l)

Value (l) = 0, Open (s) = 1

(b) Se s = n, vá para (7C) se não, deve-se fazer:

s = s + 1

l = l + 1

V = Cycle (l) e volta para o Passo (4)

(C) deixa l = l - 1 e volta para (7a)

Passo 8 - Se i = n. Pare.

O grafo não tem circuito negativo. Se não, se faz i = i + 1 e volta para (2). Se G (X, A) houver circuito negativo, no fim da computação, Cycle (j) , j =1...l são os vértices do circuito alocado.

V.3 - FLUXO COM CUSTO MÍNIMO EM UMA REDE

Para se determinar um fluxo que possua um custo mínimo, vários algoritmos foram desenvolvidos de modo que, utilizem o método primal-dual de programação linear. Estes são discutidos por FORD e FULKERSON (9).

Serão apresentados a seguir dois algoritmos que não utilizam os conceitos de programação linear, porém são muito eficientes em termos de computação.

V.3.1 - ALGORITMO DE BUSACKER E GOWEN (7)

Passo 1 - Inicia-se com todos os arcos tendo-se um fluxo igual a zero.

Passo 2 - Definir os custos modificados C^*_{ij} com respeito ao fluxo dado que existe na rede, da seguinte maneira:

$$C^*_{ij} = C_{ij} \text{ se } 0 \leq f_{ij} < b_{ij}$$

$$C^*_{ij} = \infty \text{ se } f_{ij} = b_{ij}$$

$$C^*_{ij} = - C_{ji} \text{ se } f_{ji} > 0$$

Passo 3 - Encontra-se o menor caminho ou o caminho do menor custo de N_s para N_t usando os custos modificados C^*_{ij} obtida no Passo 2. Então, transporte-se um fluxo igual ã δ onde $\delta = \min (b_{ij} - f_{ij}, f_{ji})$ ao longo deste caminho atẽ que o roteiro não seja mais o caminho de menor custo.

Substitui-se o valor de fluxo antigo pelo valor de fluxo antigo, mais o fluxo ao longo deste caminho. Se o novo fluxo tiver o valor V, Pare. Caso contrãrio, retorne ao Passo 2.

Este algoritmo tem a propriedade de que:quando o fluxo no Passo (3) ẽ P, fornece o fluxo de custo mĩnimo de P unidades de s para t. Sendo assim, obtẽm-se o fluxo de custo minimo para P = 1, 2, ... V.

Ilustração (10) - Considere-se a rede da Figura (V.3), com as respectivas demandas em cada vértice. Deseja-se enviar 2 unidades de fluxo de N_s para N_t com custo mínimo.

Passo 1 - Faz-se $f_{ij} = 0$

Passo 2 - Definir $C^*_{ij} = C_{ij}$

Passo 3 - Determinar o menor caminho de N_s para N_t, utilizando a matriz do custo modificado. Pode-se obter um desses caminhos (N_s, x_1, x_2, N_t) ou (N_s, x_3, x_2, N_t). Escolhendo o primeiro caminho, teremos:

Passo 2 - $C^*_{s1} = \infty$, conforme $f_{s1} = 1 = b_{s1} = 1$,

$C^*_{1s} = -1$

$C^*_{12} = \infty$, conforme $f_{12} = 1 = b_{12}$,

$C^*_{21} = -2$

$C^*_{2t} = \infty$, conforme $f_{2t} = 1 = b_{2t}$,

$C^*_{t2} = -1$

Para os outros arcos teremos $C^*_{ij} = C_{ij}$.

Passo 3 - Encontrando o menor caminho com custo modificado, obteremos o menor caminho (N_s, x_3, x_2, x_1, x_4, N_t) com o custo total $1 + 2 + (-2) + 2 + 2 = 5$. Transporta-se uma unidade de fluxo ao longo desse cami-

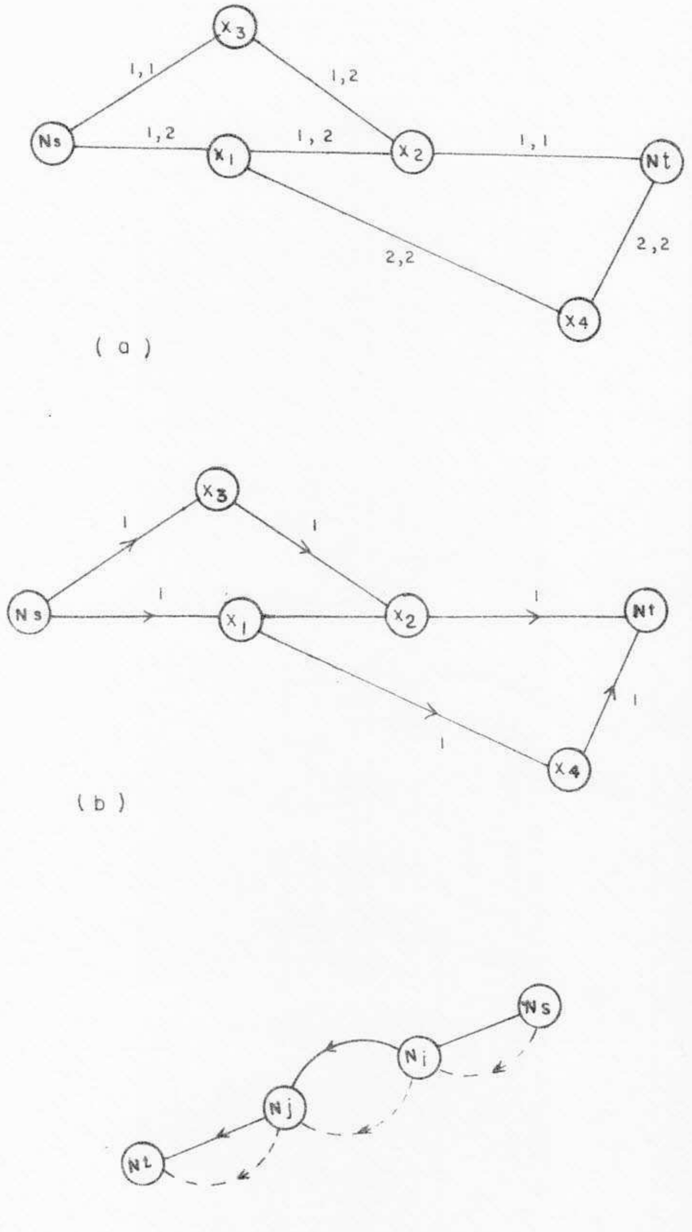


FIGURA V.3 — FLUXO COM CUSTO MÍNIMO

nho e, obtém-se os fluxos finais na rêde da (Fig. V.3.b) na qual os números indicam os fluxos em arcos.

V.3.2 - ALGORITMO DE KLEIN (11)

Passo 1 - Encontra-se algum fluxo possível de V unidades de N_s para N_t. Isto pode ser feito por tentativa, ou, então, usando-se o algoritmo para a determinação do fluxo máximo. FORD e FULKERSON (9).

Finaliza-se quando é encontrado o valor V.

Passo 2 - Definem-se os custos modificados C_{ij} como se seguem:

$$C^*_{ij} = C_{ij} \text{ se } f_{ij} < b_{ij} \; (0 \leq f_{ij})$$

$$C^*_{ij} = \infty \text{ se } f_{ij} = b_{ij}$$

$$C^*_{ij} = - C_{ji} \text{ se } f_{ji} > 0$$

Passo 3 - Usando C^*_{ij} como distância, encontre-se circuitos negativos na rede. Se não existir, o fluxo corrente é ótimo. Se existir um circuito negativo sobrepõe-se um ciclo de fluxo δ, onde $\delta = \min (b_{ij} - f_{ij}, f_{ji})$ no circuito negativo e volta-se para o Passo (2). (Se os circuitos negativos são disjuntos sobrepõe-se fluxo sobre cada um deles).

A prova de otimização depende do seguinte teorema, o qual pode ser considerado como o teorema central do fluxo do custo mínimo.

Teorema (10) - Um fluxo com valor V é ótimo, se e somente se, baseado nos custos modificados, existem somente circuitos que não sejam negativos.

Ilustração (4) - Considera-se a rede da (Figura V.4) na qual, o primeiro número nos arcos se refere à capacidade, e o segundo número ao custo. Deseja-se obter 20 unidades de fluxo com custo mínimo de s para t. Utiliza-se o algoritmo de FLOYD (7) para determinar circuitos negativos no passo (3) do algoritmo.

Passo 1 - O algoritmo de fluxo máximo produz o fluxo possível com o seguinte custo:

16 (7 + 25 + 5) + 4 (28 + 7) = 732.

Passo 2 - Dependendo a esse fluxo obtém-se a rede da Figura (V.5).

Durante a última iteração $C^4_{4,4}$ torna-se negativo com o valor (-15), indicando que existe um circuito com custo negativo, envolvendo o vértice x_4. Esse circuito pode ser obtido da matriz do custo ou seja, $(x_4, x_3, x_2. x_4)$. Tabela (V.1) e (V.2).

Passo 4 - O valor de δ será obtido como se segue:

$$\delta = \min \left[\underset{(x_4,\ x_3)}{12}, \underset{(x_3,\ x_2)}{16}, \underset{(x_2,\ x_4)}{18} \right] = 12$$

O novo modelo de fluxo depois da circulação de fluxo de δ no circuito, então o custo é de 552.

Voltando para o Passo 2, obtém-se a rede modificada.

Procedendo da mesma maneira obtém-se:

Passo 3 - Circuito (x_6, x_1, x_4, x_6) com custo - 10 é detetado

Passo 4 - $\delta = \min \left[4, 11, 19 \right] = 4.$

O novo modelo de fluxo é de custo 512

Passo 2 - A rede do novo fluxo

Passo 3 - Circuito negativo $(x_7, x_3, x_2, x_5, x_7)$ com custo -8 é investigado.

Passo 4 - $\delta = \min \left[16, 4, 16, 16 \right] = 4.$

O novo modelo de fluxo é de custo 480

Passo 2 - Obtenção da nova rede

Passo 3 - $\delta = \min (7, 15, 1, 4, 4, 16) = 1.$

O custo é igual a 476

Passo 2 - Novamente, obtém-se a rede modificada.

Passo 3 - O circuito negativo (x_1, x_2, x_4, x_1) com custo (-2) é determinado.

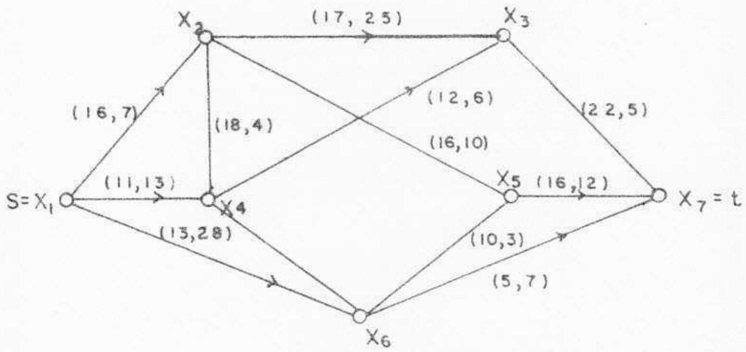


FIGURA V.4 . RÊDE PARA ILUSTRAÇÃO II

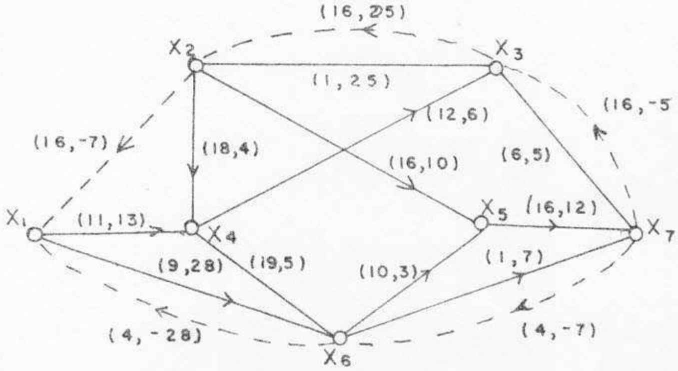


FIGURA V.5 — RÊDE ALTERADO DEPOIS DE ITERAÇÃO I

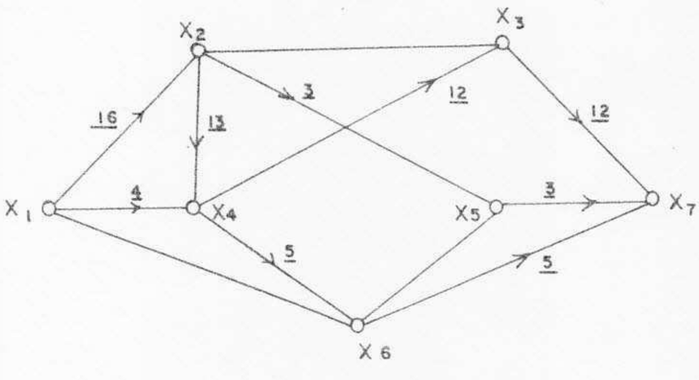


FIGURA V 6 — FLUXO COM CUSTO MÍNIMO

	X_1	X_2	X_3	X_4	X_5	X_6	X_7
X_1	0	∞	∞	13	∞	28	∞
X_2	-7	0	25	4	10	21	30
X_3	-32	-25	0	-21	-15	-4	5
X_4	-26	-19	6	-15	-9	2	11
X_5	∞	∞	∞	∞	0	∞	12
X_6	-28	∞	∞	-15	3	0	7
X_7	-37	-30	-5	-26	-20	-9	0

TABELA V.1 - MATRIZ DO MENOR CUSTO

	X_1	X_2	X_3	X_4	X_5	X_6	X_7
X_1	1	1	1	1	1	1	1
X_2	2	2	2	2	2	1	3
X_3	2	3	3	2	2	1	3
X_4	2	3	4	2	2	1	3
X_5	5	5	5	5	5	5	5
X_6	6	6	6	6	6	6	6
X_7	2	3	7	2	2	1	7

TABELA V.2 - MATRIZ DE CAMINHO

Passo 4 - S = (1, 6, 5) = 1.

Agora o custo é de 474 (Fig. V.6).

Na rede modificada não se pode achar circuito negativo, portanto o menor custo para 20 unidades de fluxo é de 474.

Na Figura (V.7) a linha que se encontra em baixo dos números indicam o fluxo ao longo do arco correspondente.

V.3.3 - ALGORITMO DA DESIGNAÇÃO

Para se obter a designação de custo mínimo, as seguintes operações devem ser realizadas:

Passo 1 - Subtrair o elemento mínimo de cada linha de todos os elementos daquela linha. Fazer o mesmo para com as colunas.

Passo 2 - Examinar as linhas e colunas sucessivamente, para cada linha (coluna) com exatamente um zero restante, reservar aquela posição para uma designação, e eliminar (x) os outros zeros da coluna (linha) correspondente. Repetir, se necessário, para as linhas e colunas sem posições reservadas, completando as designações, a solução é ótima. Caso contrário, seguir para o passo (3).

Passo 3 - Traçar um número mínimo de rotas para cobrir todos os zeros, da seguinte maneira:

a) Marcar todas as linhas que não tenham designações.

b) Marcar todas as colunas que tenham designações em colunas.

c) Marcar todas as linhas que tenham designações , em colunas marcadas.

d) Repetir os passos (b) e (c) até não ser mais possível marcar linhas ou colunas.

e) Traçar uma reta sobre cada linha não marcada e, sobre cada coluna marcada.

Passo 4 - Examinar todos os elementos não cobertos por uma reta. Escolher o elemento mínimo desses elementos e subtraí-lo de todos os elementos não cobertos por uma reta. Somar esse elemento mínimo a cada elemento situado na interseção de duas retas. Retornar ao passo (2).

Ilustração (14) - Desejando-se designar 3 equipamentos aos 4 postos de modo que se minimize o tempo total de deslocamento, procede-se da seguinte maneira:

É necessário criar um equipamento fictício x_4 com todos os custos nulos, para igualar o número de equipamento com o número de postos disponíveis. Isto feito na Tabela (V.3), obter-se-á matriz da tabela (V.4).

Aplicando-se os Passos (1), (2), (3) do algoritmo a Tabela (V.5) será obtida.

O elemento mínimo não coberto é igual a 2. Aplicando-se o Passo (4) a Tabela (V.5) se transforma em Tabela (V.6).

Ao se aplicar o Passo (5) na Tabela (V.6) chegou-se ao ponto em que se faz necessário arbitrar uma designação pelo fato das linhas e colunas apresentarem mais de um zero disponível para a designação. Isto indica a existência de mais de uma solução ótima. Escolhendo, arbitrariamente , uma designação (x_1, y_2) obtém-se a matriz da Tabela (V.7). As designações ótimas serão vistas na Tabela (V.8) em que as duas soluções (1 e 2) são ótimas (com o mesmo tempo de deslocamento).

	Y_1	Y_2	Y_3	Y_4
X_1	5	1	3	100
X_2	3	1	4	3
X_3	3	3	4	2

TABELA V.3 - MATRIZ INICIAL

	Y_1	Y_2	Y_3	Y_4
X_1	5	1	3	100
X_2	3	1	4	3
X_3	3	3	4	2
X_4	0	0	0	0

TABELA V.4 - MATRIZ DE EFICIÊNCIA

	Y_1	Y_2	Y_3	Y_4
X_1	4	$\boxed{0}$	2	99
X_2	2	$\cancel{0}$	3	2
X_3	1	1	2	$\boxed{0}$
X_4	$\boxed{0}$	0	0	0

TABELA V.5 - MATRIZ DE EFICIÊNCIA DEPOIS
DA PRIMEIRA ITERAÇÃO

	Y_1	Y_2	Y_3	Y_4
X_1	2	$\boxed{0}$	$\cancel{0}$	97
X_2	0	0	1	$\cancel{0}$
X_3	1	3	2	$\boxed{0}$
X_4	$\cancel{0}$	2	0	$\cancel{0}$

TABELA V.6 - MATRIZ DE EFICIÊNCIA ALTE-
RADA NA SEGUNDA ITERAÇÃO

	Y_1	Y_2	Y_3	Y_4
X_1	0	0	$\cancel{0}$	97
X_2	$\boxed{0}$	$\cancel{0}$	1	$\cancel{0}$
X_3	1	3	2	$\boxed{0}$
X_4	$\cancel{0}$	2	$\boxed{0}$	$\cancel{0}$

TABELA V.7 - DESIGNAÇÃO ÓTIMA NA MATRIZ
DE EFICIÊNCIA

	1	2
EQUIPAMENTO	POSTO	POSTO
X_1	Y_2	Y_3
X_2	Y_1	Y_2
X_3	Y_4	Y_4

TABELA V.8 - DESIGNAÇÃO ÓTIMA

CAPÍTULO VI

DESCRIÇÃO DOS MÉTODOS DE RESOLUÇÃO

VI.1- INTRODUÇÃO

Como visto no Capítulo II, o objetivo deste trabalho é otimizar o tempo de deslocamento dos equipamentos da pesquisa nas diferentes etapas. Neste Capítulo serão apresentados os três métodos, pelos quais se resolveu o problema. São apresentadas, inicialmente, duas maneiras, nas quais não são utilizados os conceitos de programação linear, mas que são eficientes em termo de computação. A terceira maneira foi resolver o problema utilizando o algoritmo da designação.

Antes de desenvolver os métodos de resolução, será conveniente tecer algumas considerações referentes à resolução do problema.

VI.2 - HIPÓTESES

Considerando as restrições existentes na pesquisa feita pela SUDENE, definiremos cinco redes, tendo como os seus vértices, os postos da Coleta de Dados em cada etapa distribuída da seguinte maneira:

Se um equipamento não foi utilizado na próxima etapa (por exemplo; nas épocas de festas em que o movimento não é normal, ou, então, porque geralmente cada grupo orientava dois equipamentos, e não se podia transportar somente um deles enquanto o outro equipamento estivesse ocupado) então, este não foi considerado. Explica-se que esses equipamentos serão utilizados como um centro de suprimento nas próximas etapas.

Como foi dito anteriormente, a cidade de Campina Grande é o centro da distribuição dos equipamentos, isto é: todos os equipamentos foram enviados de Campina Grande e depois do término da coleção total dos dados voltarão para a mesma. Portanto, nas etapas em que houve menos de vinte e cinco equipamentos a serem distribuídos, considerou-se um Posto de Suprimento em Campina Grande, cuja capacidade varia, dependendo dos equipamentos localizados aqui e, também, das demandas da próxima etapa.

A Tabela de relação de grupos, postos e equipamentos (Tabela II.1) mostra que existem alguns postos que têm uma necessidade maior que a dos outros e, portanto, precisam de mais de um equipamento. É óbvio que o número de equipamentos em Campina Grande varia de acordo com as etapas.

Desde que, cada arco na rede tenha diferentes capacidades, precisaríamos armazená-las no computador e, em alguns casos, isto se tornaria impossível devido às limitações da memória. Por isso, foi necessário e mais fácil considerar que a capacidade de todos os arcos seja igual a 1 (um) e, para isso, se se tiver um posto com capacidade maior que um, este seria substituído por um conjunto de postos, igual a sua capacidade. Tal critério foi adotado devido ao fato de não haver muitos postos com capacidade maior que um.

Verificou-se que o problema é satisfazer as demandas ao custo mínimo de m_t destinos pelo n_s origens que pode ser convertido a um problema de achar um fluxo máximo com o custo mínimo.

Se adicionarmos um vértice artificial de origem (s) e um vértice artificial de destino (t), desde que esses não tenham sentido físico, os custos nos arcos de s para n_s e nos arcos de m_t para t serão nulos. As Figuras (VII.1 a VII.5) representam os grafos ultimamente obtidos.

Visto que, o objetivo é satisfazer a um custo mínimo, as demandas da seguinte etapa em cada rede, então, considerou-se que não existe nenhum arco de uma origem para outra e nem de um destino para outro. Os custos nos arcos que unem cada origem a todos os destinos são os menores possíveis (veja Capítulo IV). A capacidade nesses arcos que unem s a n_s é igual a um. Desde que se deseja transportar um fluxo com o custo mínimo, então o problema pode ser resolvido pelos métodos indicados na próxima Secção.

VI.3 - MÉTODOS DE RESOLUÇÃO

VI.3.1 - MÉTODO DE BUSACKER E GOWEN (3)

O algoritmo deste método foi apresentado no Capítulo IV. O segundo passo do algoritmo exige determinar o menor caminho entre s e t. O problema foi resolvido, uma vez utilizando-se o algoritmo de MOORE (13), BELLMAN (1) e FORD (8) para achar o menor caminho entre s e t. Utilizando-se o vetor θ que contém os vértices do menor caminho, modificou-se os custos associados aos arcos do menor caminho e, uma vez que o fluxo é igual à capacidade, para se obter a matriz de custo modificado, as seguintes alterações foram feitas:

$$C^*_{ij} = \infty \quad (\text{conforme} - f_{ij} \text{ é igual a } b_{ij})$$

$$C^*_{ji} = -C_{ij} \, (\text{conforme} - f_{ij} \text{ é positivo})$$

O processo foi feito V vezes para obter o fluxo máximo (V) com o menor tempo de deslocamento. Finalmen te obtemos o tempo total de viagem da seguinte maneira:

Se na solução final tiver um fluxo positivo no arco de m_j para n_i, isto significa que a demanda de m_j será satisfeita por n_i. Somando-se essas quantidades obteve-se o tempo total de viagem em cada rede. (Apêndice A)

O processo foi feito outra vez, utilizando-se o algoritmo de FLOYD (7), pelo qual pôde-se obter o menor caminho entre todos os pares de vértices, por conseguinte, o menor caminho entre s e t. Os outros processos foram explicados anteriormente.

Para determinar o menor caminho entre s e t, nota-se que a deficiência do algoritmo de FLOYD (7) é inferior a do algoritmo de FORD (8), BELLMAN (1) e MOORE (13). Porque despende de maior tempo na execução, maior uso da memória do computador devido ao método de pesquisa.

VI.3.2 - MÉTODO DE KLEIN (11)

Explicou-se no Capítulo IV o método apresentado por KLEIN. Deve-se enviar inicialmente, um fluxo máximo, o que pode ser feito por tentativa, ou então, utilizando-se o algoritmo de fluxo máximo (9). Transportou-se o fluxo máximo de maneira que, cada origem satisfaça a um só destino, e cada destino não receba o fluxo mais de uma vez. Em outras palavras:

$$\sum_{i=1}^{n} a_i = 1 \quad e \quad \sum_{j=1}^{n} b_j = 1$$

Portanto, definiu-se a matriz do tempo alterada da seguinte maneira:

$$C^*_{ij} = \infty \quad \text{conforme } f_{ij} \text{ é igual à } b_{ij}$$

$$C^*_{ij} = -C_{ij} \quad \text{conforme } f_{ij} \text{ é positivo}$$

Para verificar se existe um circuito com custo negativo na rede ou não, utilizou-se o método da pesquisa direita (6) e, também, o método de FLOYD. Se existir um circuito negativo, sobrepõe-se um ciclo de fluxo δ, onde $\delta = \min (b_{ij} - f_{ij}, f_{ji})$ no circuito negativo. Nesse caso o valor de δ é igual a 1 (um). Continua-se dessa maneira até que não se pode encontrar um circuito negativo na rede.

Para detetar circuito negativo, o método da pesquisa direta (6) oferece bons resultados com pequenas dimensões, mas com grandes dimensões (50 x 50) os primeiros circuitos negativos são detetados rapidamente, mas quando estamos bem próximos da solução ótima essa operação é muito demorada. Por que esse algoritmo utiliza a técnica de "back truck" com complexidade exponencial.

O método de FLOYD utilizado para achar circuito com custo negativo, se for usado para uma rede em que a pesquisa parte do vértice s a todos os outros vértices, não é o melhor método. Uma técnica mais eficiente é o algoritmo de FORD (8), MOORE (13) e BELLMAN (1). (Apêndice B)

Na comparação dos métodos de KLEIN, BUSACKER e GOWEN visto que pelo método de BUSACKER e GOWEN (3). O número de iterações é conhecido, o qual, para o método de

KLEIN (11) é desconhecido e depende mais do fluxo inicial em que começou o processo.

VI.3.3 - MÉTODO DE DESIGNAÇÃO ÓTIMA (14)

Os problemas de designação ocorrem quando se tem de distribuir uma determinada quantidade de ítens em uma quantidade igual de localização. Nos grafos definidos nas Figuras VII.1 a VII.5, essa igualdade não se verifica, portanto, isto foi obtido com a introdução de ítens ou localizações fictícias, conforme necessário. (Apêndice C)

Os resultados serão vistos no Capítulo (VI). Em alguns casos mais de uma solução ótima foi obtida, a qual é mostrado nas Figuras (VII.1 a VII.5) pelas retas tracejadas.

Se considerarmos n o número de equipamentos disponíveis e m a demanda nos postos em cada rede, a matriz de custo para os métodos de Busacker e Gowen (7) e o de KLEIN é de ordem (n + m + 2) x (n + m + 2) o qual para algoritmo de designação é de (n x n) ou (m x m) dependendo do que for maior (m ou n).

VI.3.4 - MÉTODO DE CLARKE E WRIGHT (5)

Os métodos vistos nos ítens (VI.3.1),(VI.3.2) e (VI.3.3) são utilizados quando diferentes regiões na Coleta de Dados foram definidos mas pode-se otimizar o tempo muito mais ainda. A seguir veremos um modelo relativamente simples e fácil de resolver.

No ano de 1963 CLARKE e WRIGHT (10) introduziram o conceito de "economia" e desde aquela época muitos métodos que tinham sido publicados, tentam resolver o problema de planejamento dos veículos, usando técnicas baseadas naquele conceito.

Considera-se um armazém para abastecer n consumidores e, além disso, supõe-se que exista bastante veículos com a capacidade suficiente no depósito, de maneira que, cada consumido pode ser abastecido por um veículo e vice-versa (Fig. V.1). O custo total (ou distância percorrida por todos os veículos é:

$$2 \sum_{j=1}^{n} C_{oj}$$

onde o sufixo nulo representa o depósito, e C_{oj} é o custo de viagem do depósito até o consumidor j. Nesse caso o número de veículos usado é n. Agora supõe-se dois consumidores i e j unidos, de modo que sejam fornecidos por um só veículo e numa rota só. A ligação entre consumidores i e j elimina uma rota do veículo e, também, reduz o custo por quantidade $S_{ij} = C_{oi} + C_{oj} - C_{ij}$. A quantidade S_{ij} é chamada de a economia na ligação (ij) que não é negativa; é óbvio que quanto maior a economia entre o consumidor i e o consumidor j, a união entre os consumidores i e j numa rota é a mais desejada.

- Algoritmo de CLARKE e WRIGHT

Esse algoritmo pode ser descrito pelos seguintes passos:

(a) Calcular as S_{ij} para todos os pares de consumidores ij.

(b) Ordenar as S_{ij} na forma descendente de grandeza.

(c) Começar de cima da lista, como se segue:

(i) se a produção de um aro for possível de acordo com as restrições do problema, adicionar essa ligação à solução; se não despreza-se o aro (Nesse caso as seguintes restrições devem ser consideradas:

- cada rota deve abastecer 4 Postos

- Precisa-se de 23 rotas para abastecer 92 Postos).

(ii) Verifica-se o aro seguinte na lista e repete-se (i) até que a lista seja esgotada.

(d) Os aros que tinham sido escolhidos organizam a solução para o problema.

Utilizando-se esse método, pode-se obter vinte e tres rotas, cada uma abastecendo quatro postos de coleta dos dados.

Esse algoritmo é heurístico e pode se obter uma boa solução que geralmente é a solução ótima. Esse método é muito simples de ser aplicado e precisa somente de muito pouco tempo de computador para a execução. Contudo, algumas restrições são difíceis de serem incorporadas, e a necessidade de armazenagens no computador pode tornar-se excessiva.

Ilustração - Supõe-se que existem dois veículos no armazem (o) para abastecer seis consumidores.

Inicialmente calculam-se as economias (S_{ij}) para todos os pares de consumidores, e se as ordenam na forma descendente. (Tabela VI.1)

As duas rotas são mostradas na Figura VI.2. A distância percorrida na rota (0,1,2,3,0) é igual a 13 km e, na outra rota (0,4,5,6,0) é igual a 18.5 km.

Observa-se que qualquer restrição aumentará a distância percorrida pelos veículos.

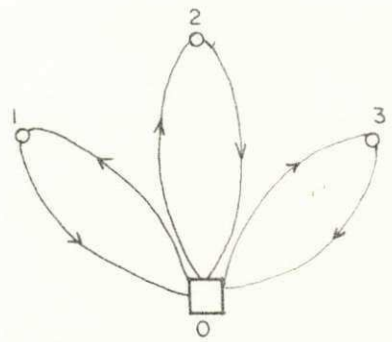


FIGURA VI.1 - ABASTECER OS CONSUMIDORES COM BASTANTE VEÍCULO DO ARMAZEM O

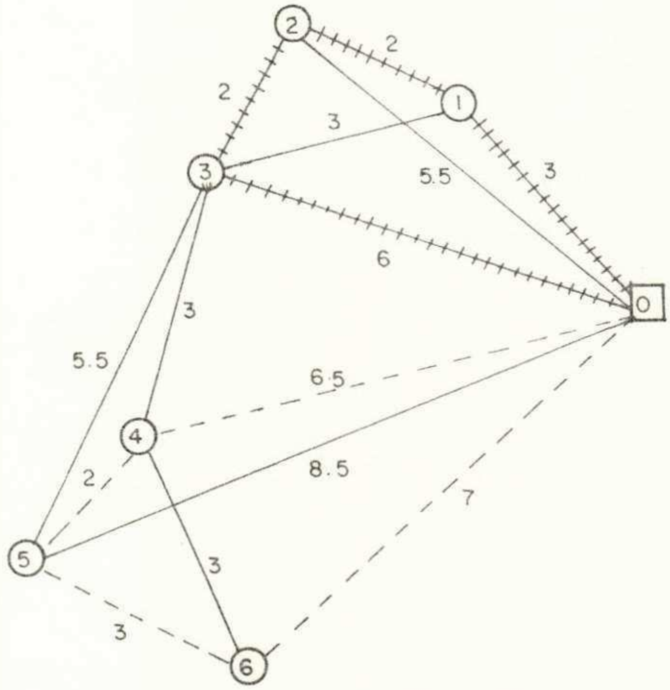


FIGURA VI. 2 - OBSTENÇÃO DAS RETAS ÓTIMAS SEM CONSIDERAR NENHUMA RESTRIÇÃO

TABELA VI.1

S_{34}	S_{56}	S_{34}	S_{23}	S_{35}	S_{12}	S_{13}	-----
13.0	12.5	10.5	9.5	9.0	6.5	6.0	----

A TABELA DE Sij ORDENADA

CAPÍTULO VII

APRESENTAÇÃO DE RESULTADOS

Este Capítulo tem a finalidade de ilustrar o uso dos dados obtidos como resultado dos programas desenvolvidos durante esta otimização. Os dados produzidos são de natureza repetitiva e, em grande posto, escolheu-se apenas um exemplo como ilustração. (Apêndice D)

Os resultados obtidos referentes aos diferenres passos da pesquisa são mostrados nas Fig. (VII.1) a (VII.5), nas quais os arcos identificados mostram o arranjo ótimo. No caso em que houve mais de uma solução ótima mostrou -se pelas linhas tracejadas.

A Tabela (VII.1) mostra a comparação entre o arranjo real aplicado na coleta de dados e o arranjo sujerido neste trabalho em cada passo. Analisando estes resultados observou-se 22.5% de economia no tempo total de viagem.

Observando a Tabela (VII.1) vemos que na primeira etapa não se verificou qualquer economia de tempo. Isto se explica pelo fato de que a rede é pequena e por os pesquisadores haverem escolhido o caminho ótimo. No caso da etapa 4, na qual verifica-se uma economia de 35.9% significa que, em razão de existir muitas opções de escolha, tornou-se difícil escolher o caminho ótimo e, com isso se distanciaram da solução ótima encontrada neste trabalho. Se existisse mais postos de coleta de dados por exemplo: 50 a ser abastecido em cada etapa verificar-se-ia que se tornava ainda mais difícil escolher as rotas ótimas nas quais a importância de pesquisa operacional seria observada mais ainda.

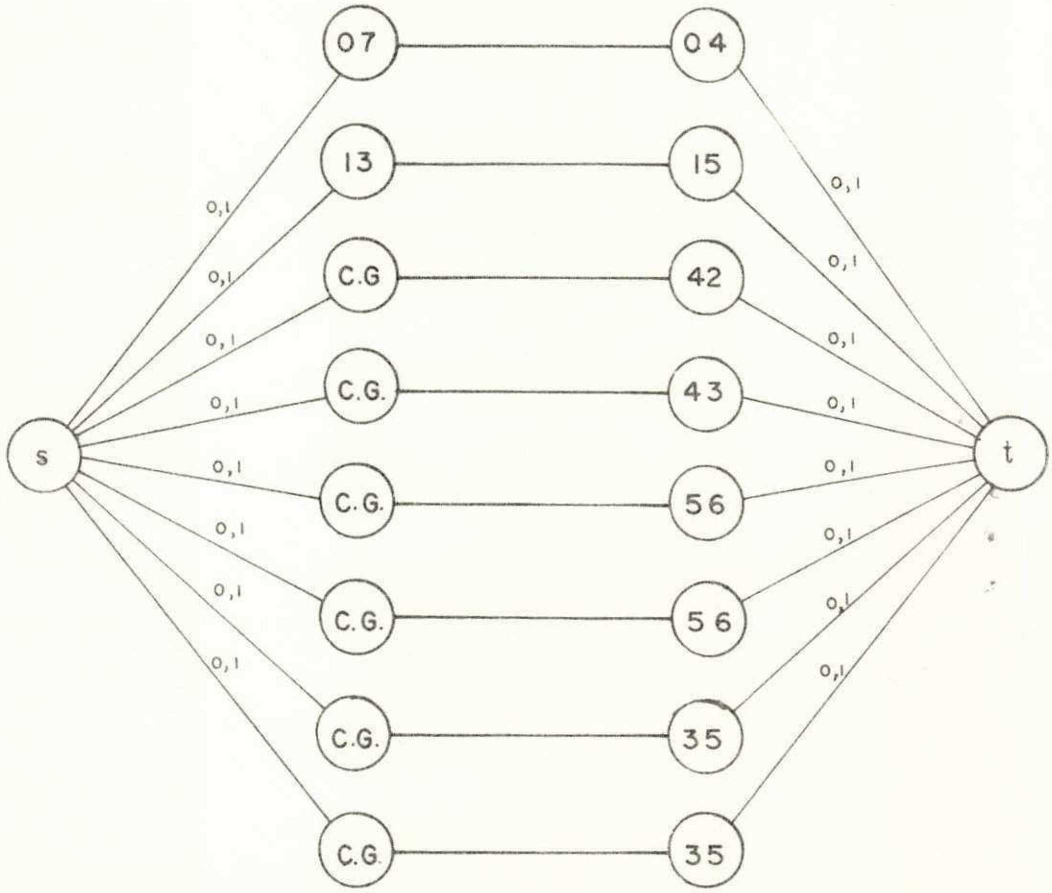


FIGURA VII.1 - RÊDE ORGANIZADA PELAS ETAPAS 1 e 2 DE COLETA DE DADOS E A SOLUÇÃO ÓTIMA

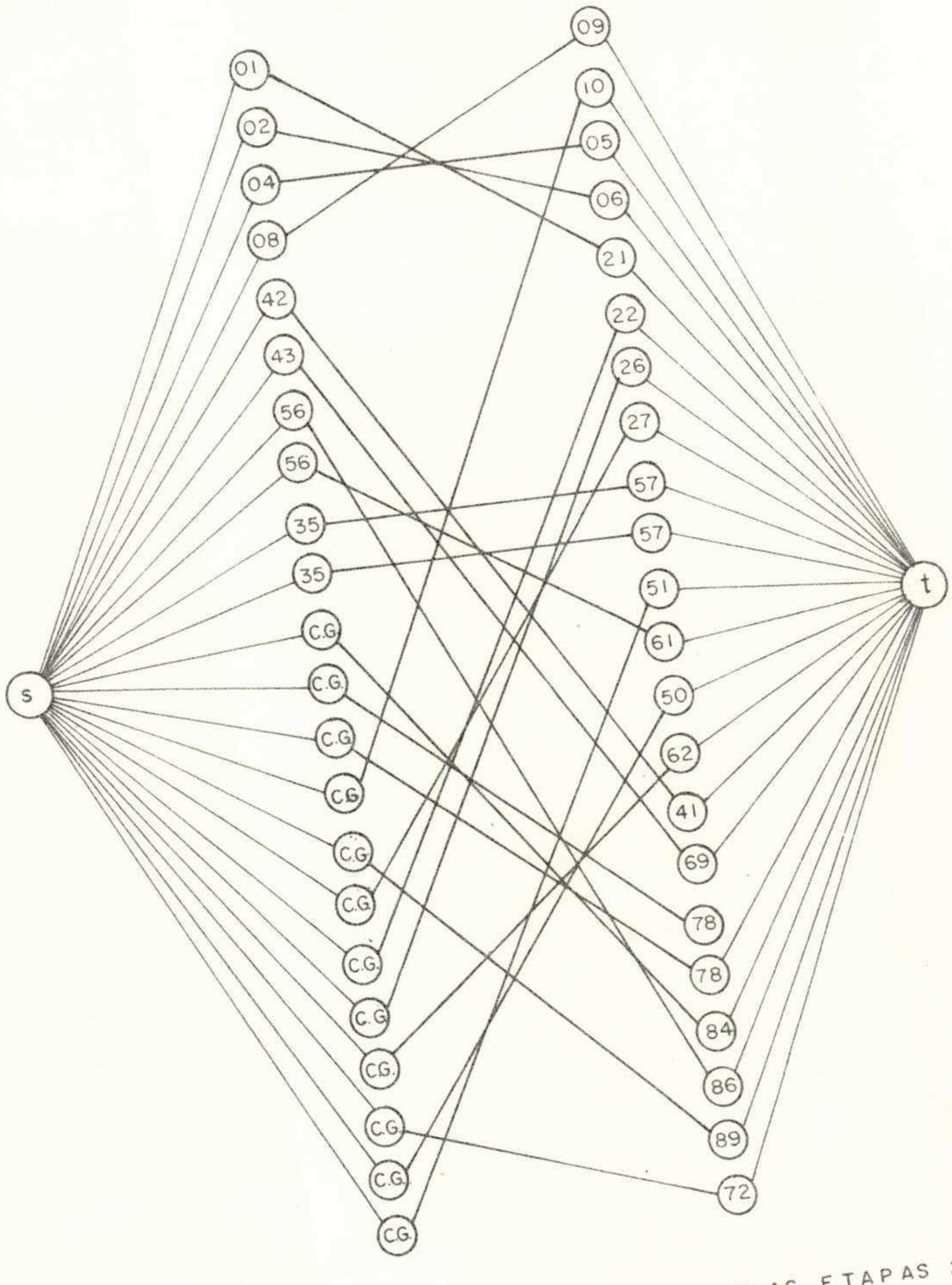


FIGURA VII.2 – RÊDE ORGANIZADA PELAS ETAPAS 2 e 3 DE COLETA DE DADOS E A SOLUÇÃO ÓTIMA

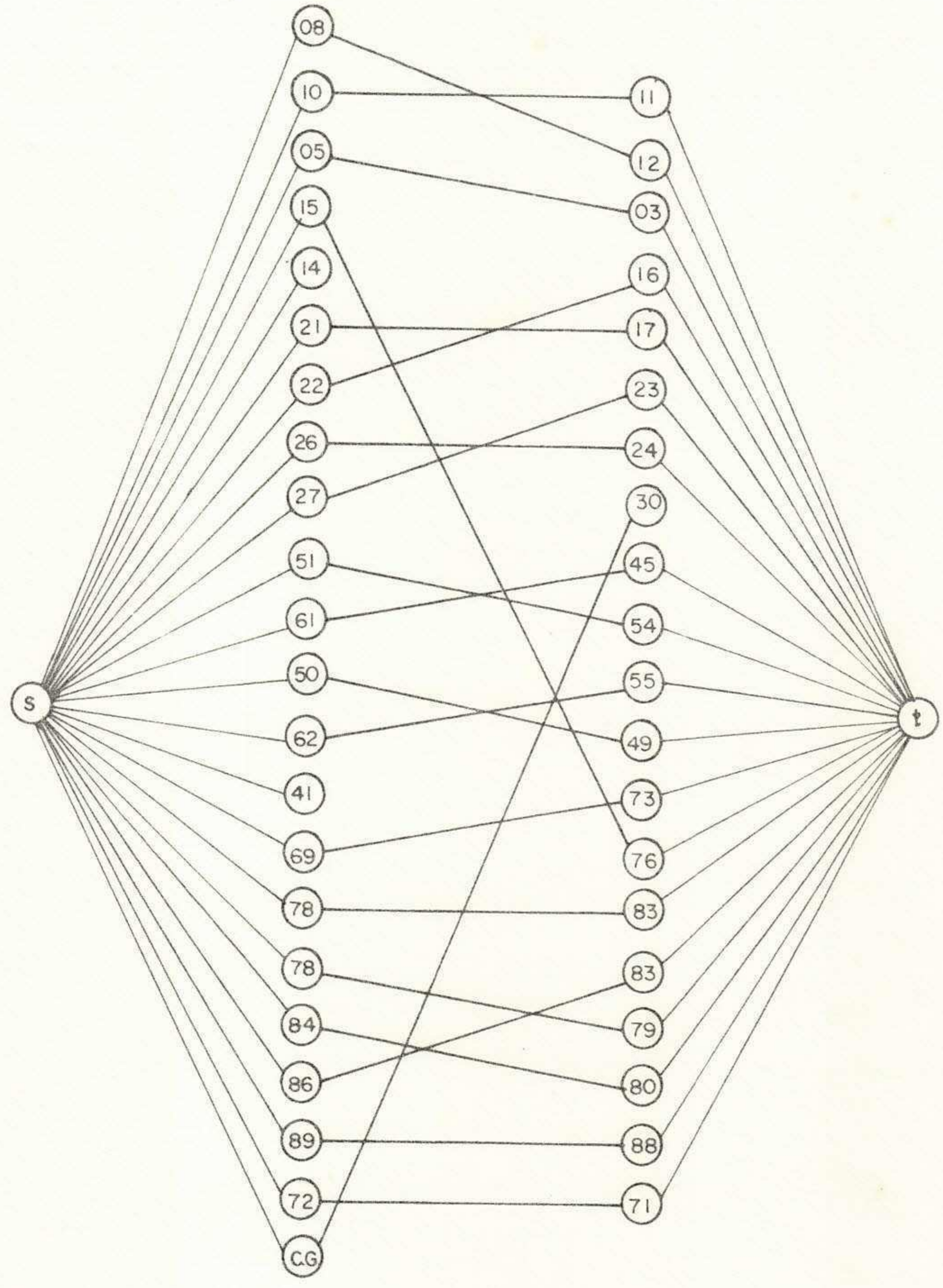


FIGURA VII.3 — RÊDE ORGANIZADA PELAS ETAPAS 3 e 4 DE COLETA DE DADOS E A SOLUÇÃO ÓTIMA

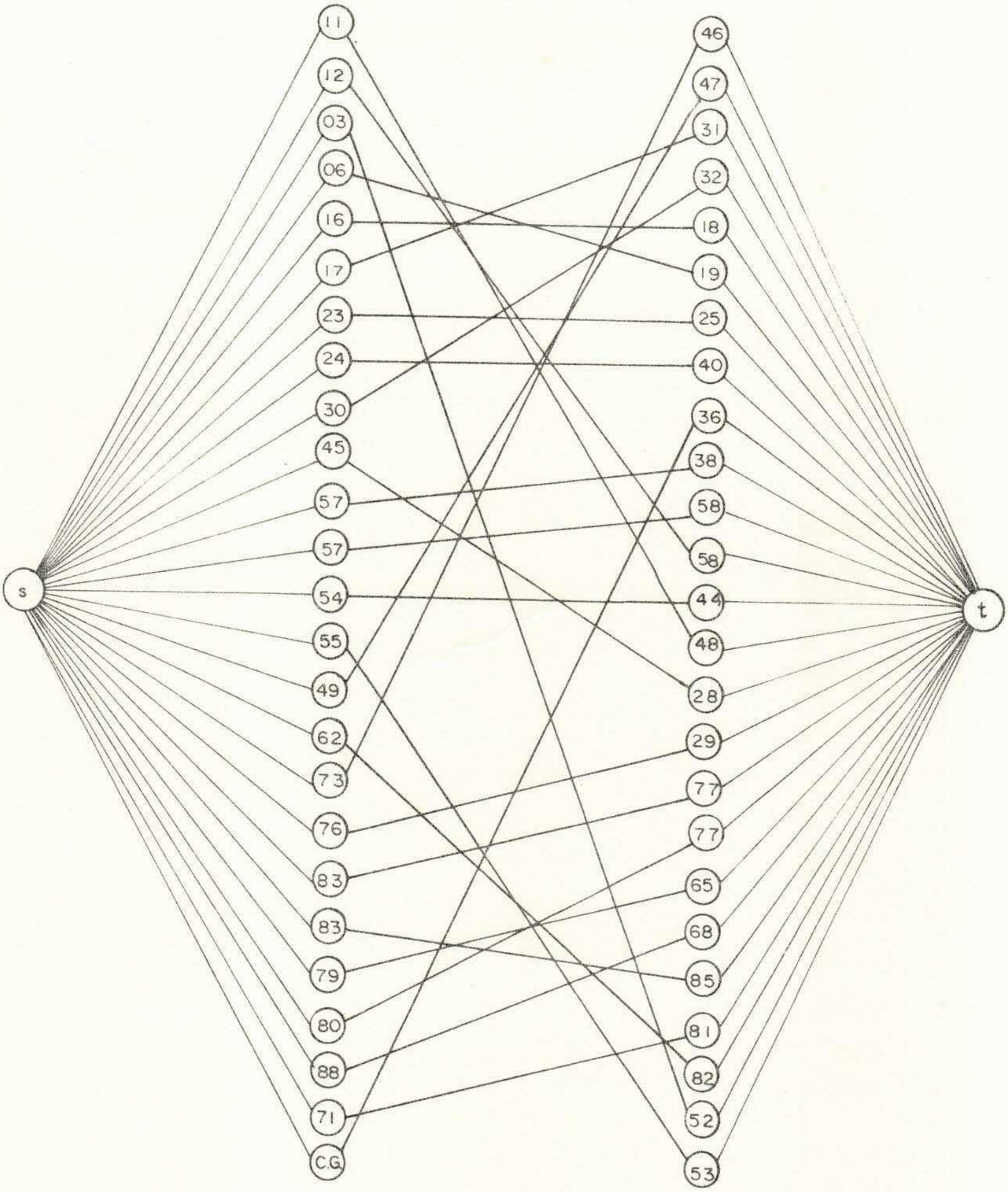


FIGURA VII.4 - RÊDE ORGANIZADA PELAS ETAPAS 4 e 5 DE COLETA DE DADOS E A SOLUÇÃO ÓTIMA

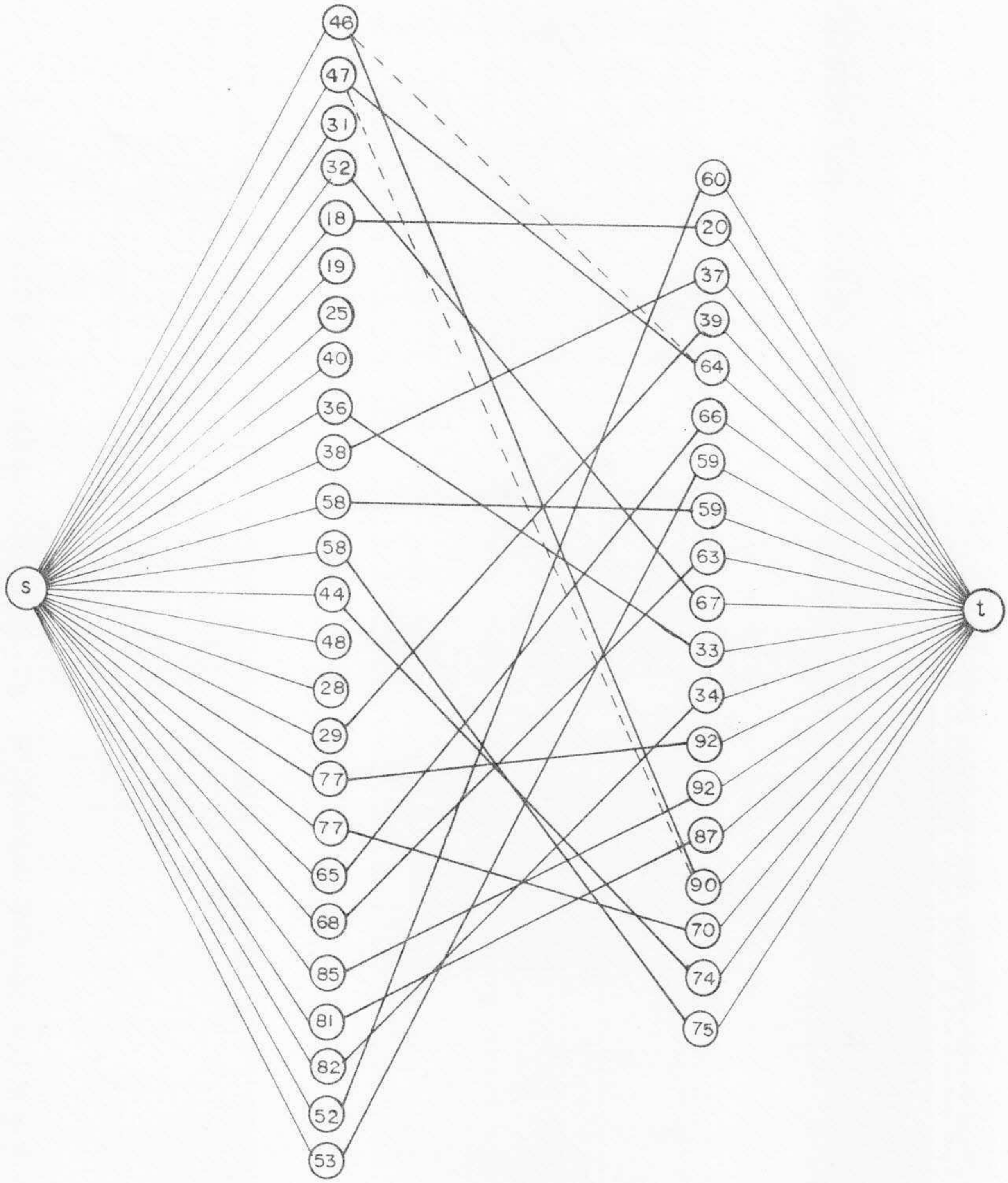


FIGURA VII.5 – RÊDE ORGANIZADA PELAS ETAPAS 5 e 6
E SOLUÇÃO ÓTIMA

7

	ARRANJO REAL APLICADO	ARRANJO OFERECIDO	POR CENTO DE ECONOMIA
1ª Parte	31.13	31.13	-
2ª Parte	177.56	166.31	6.4 %
3ª Parte	65.96	44.72	3.2 %
4ª Parte	132.96	85.50	35.7 %
5ª Parte	113.72	72.88	35.9 %
T O T A L	516.79	400.54	22.5 %

TABELA VII.1 - COMPARAÇÃO ENTRE ARRANJO APLICADO E O ARRANJO SUGERIDO

CAPÍTULO VIII

C O N C L U S Õ E S

Este trabalho tratou do desenvolvimento e da aplicação da Teoria dos Grafos para a Otimização do Tempo de Viagem ou deslocamento de equipamento entre vários polos. O modelo foi satisfatoriamente ensaiado sobre o deslocamento de equipamento de coleta de dados feita na Região do Nordeste Brasileiro.

Considerando todas as restrições do caso real e, aplicando-se o modelo conseguiu-se uma economia em tempo total de viagem de até 35.9% em alguma etapa e, também, uma média de 22.5%.

Se realmente as divisões de Regiões no Processo da Coleta de Dados não são necessárias, ou, então, somente existe uma Região, na qual a Pesquisa deve ser feita inicialmente (Utilizando o Método de CLARKE e WRIGHT (5)) pode-se obter melhores resultados. De fato,deve se modficiar o algoritmo de CLARKE e WRIGHT para se adaptar às condições reais da Coleta de Dados.

REFERÊNCIAS BIBLIOGRÁFICAS

(1) BELLMAN, R. (1958). On a routing problem, Quart. of Applied Mathematics, 16, P.87.

(2) BERRY, R. C. (1971), A constrained shortest path algorithm, Paper presented at the 39 th National ORSA Meeting. Dallas, Texas.

(3) BUSACKER, R.G., AND GOWEN, P.J. (1961). A Procedure for Determining a Family of Minimal - Cost Network Flow Pattern, ORD Technical Report 15, Operations Research Office, Johns Hopkins University.

(4) CHRISTOFIDES, N. (1975). Graph Theory, An Algoritmic Approach, Academic Press New York.

(5) CLARKE, G. AND WRIGHT, J. W. (1963). Scheduling of vehicles from a Central depot to a number of delivery points. ops. Res. 11, 568.

(6) FLORIAN, M., AND ROBERT, P. (Jan. 1971). A direct search method to locate negative cycles in a graph, Management Science vol. 17, Nº 5.

(7) FLOYD, R. W. (1962). Algorithm 97 - Shortest path, Comm. of ACM, 5, P. 345.

(8) FORD, L.R., JR. (1946), Network flow theory, Rand Corporation Report P.923.

(9) FORD, L.R., Jr. and Fulkerson, D.R. (1962), Flows in Networks, Princeton University Press, Princeton,N.J.

[10] HU, T.C. (1969), Integer Programming and Network Flows, Addison - Wesley Publishing Company. London.

[11] KLEIN, M. (Nov. 1967). A Primal Method for Minimal Cost Flows, Man. Sci., 14(3), 205-220.

[12] MARCONDES, L.C. (1976). Relatório Técnico do Convênio SUDENE/ATECEL, Campina Grande.

[13] MOORE, E.F. (1957). The shortest path through a maze, Proc. Int. Symp. on the Theory of Switching, Part II, P. 285.

[14] PUCCINI, A.L. (1976). Introdução à Programação Linear, Livros Técnicos e Científicos Editora S.A., Rio de Janeiro, Brasil, 1976.

[15] YEN, J.Y. (1977). On the efficiencies of algorithms for detecting negative loops in networks, Santa Clara Business Review, P.52.

APÊNDICE A

Programa 1 - Método de Busacker e Gowen utilizando-se o Algorítmo de Ford, Bellman, Moore para determinar o menor caminho de s para t.

```
      SUBROUTINE GENRAL
      INTEGER LF(53),LS(53),C(53,53),GINV(53,53),GAMAS(53) ,SS(53),SUM(
     *53),XX(53),TETA(53),P,LIST(53)
      COMMON N,C,TETA,IS,GINV,KEY
      KEY=0
      M=N
      NNN=N+1
      LF(IS)=0
      LS(IS)=0
      KEY=0
      KEY1=0
      K=1
      NCUIT=0
      ILL=IL+1
      DO 4 I=1,N
      IJ=0
      DO 3 J=1,N
      IF(I.EQ.J) GO TC 3
      IF(C(J,I).EQ.999999) GO TO 3
      IJ=IJ+1
      GINV(I,IJ)=J
    3 CONTINUE
      GINV(I,IJ+1)=NNN
    4 CONTINUE
      DO 2 II=1,M
      TETA(II)=0.
      SUM(II)=0.
      XX(II)=0.
      GAMAS(II)=0.
    2 CONTINUE
      J=0
      II=1
      DO 20 I=2,N
      IF(C(II,I).EQ.999999) GO TO 10
      J=J+1
      SS(J)=I
      TETA(I)=1.
      LF(I)=C(IS,I)
      LS(I)=LF(I)
      GO TC 20
   10 LF(I)=999999
      LS(I)=LF(I)
   20 CONTINUE
   22 IM=1
      IMM=IM
      DO 35 NM=1,J
      P=SS(NM)
      DO 30 IJ=1,M
      XX(IJ)=0.
      SUM(IJ)=0.
      KEY1=0
      IF(P.EQ.IJ) GO TO 30
      IF(C(P,IJ).EQ.999999) GO TO 30
      DO 28 MD=1,IMM
      IF(GAMAS(MD).EQ.IJ)GO TO 27
      GO TC 28
   27 KEY1=1
      GO TC 29
```

```
   28 CONTINUE
   29 IF(KEY1.EQ.1) GC TO 30
      GAMAS(IM)=IJ
      IM=IM+1
      IMM=IM-1
   30 CONTINUE
   35 CONTINUE
      KK=K+1
      DO 90 I=1,IM
      IP=C
      IX=GAMAS(I)
      IF(IX.EQ.0) GO TO 90
      DO 46 IJ=1,J
      DO 45 JI=1,M
      IF(GINV(IX,JI)-NNN) 38,46,38
   39 IF(GINV(IX,JI).EQ.SS(IJ))GO TO 40
      GO TC 45
   40 IP=IP+1
      XX(IP)=SS(IJ)
      GO TC 46
   45 CONTINUE
   46 CONTINUE
      IF(IP.GT.0) GO TO 55
      WRITE(6,49)
      GO TC 90
   49 FORMAT(10X,'IP=0'/)
   55 DO 60 IK=1,IP
      IKK=IK
      JX=XX(IK)
      SUM(IK)=LF(JX)+C(JX,IX)
   60 CONTINUE
      KL=XX(1)
      SUM1=SUM(1)
      IF(IKK.LT.2)GO TC 75
      DO 70 IL=2,IKK
      IF(SUM(IL).GT.SUM1)GO TO 70
      KL=XX(IL)
      SUM1=SUM(IL)
   70 CONTINUE
   75 IF(SUM1.GE.LF(IX)) GO TO 80
      LS(IX)=SUM1
      TETA(IX)=KL
      GO TC 90
   80 LS(IX)=LF(IX)
   90 CONTINUE
  103 FORMAT(10X,I7)
   91 NN=N-1
      IF(K-NN)110,105,105
  105 KEY=1
  110 J=0
      DO 115 LL=1,N
      IF(LS(LL).EQ.LF(LL)) GO TO 115
      NCUIT=1
      J=J+1
      SS(J)=LL
  115 LF(LL)=LS(LL)
      IF(NCUIT.EQ.0) GC TC 140
      IF(KEY.EQ.1)GO TC 140
```

```
  116 K=K+1
      NCLIT=0
      DO 120 NI=1,N
      LIST(NI)=0
  120 GAMAS(NI)=0
      GO TO 22
  140 RETURN
      END
```

```
      INTEGER C(53,53),TETA(53),TEST(53),TEMP,CC(53,53),LIST(25,2),
     *GINV(53,53),COST
      COMMON N,C,TETA,IS,GINV,KEY
      CFV=0.
      KK=0
      READ(5,102)IC,IL,IS,IVALUE
      N=IC+1
      DO 30 I=1,N
      TEST(I)=0.
      DO 30 J=1,N
      IF(I.EQ.J) GO TO 22
      C(I,J)=999999
      GO TO 30
   22 C(I,J)=0
   30 CONTINUE
      ILL=IL+1
      DO 4 II=2,IL
      READ(5,21)(C(II,I),I=ILL,IC)
    4 CONTINUE
   21 FORMAT(8I4)
      DO 40 I=1,IL
      C(1,I)=0
      C(IL+I,N)=0
   40 CONTINUE
      DO 10 I=1,N
      WRITE(6,33)I
      WRITE(6,34)(C(I,JJ),JJ=1,N)
      DO 10 J=1,N
   10 CC(I,J)=C(I,J)
    3 IF(CFV.EQ.IVALUE) GO TO 200
      CFV=CFV+1
      DO 60 I=1,N
      DO 60 J=1,N
      GINV(I,J)=0
   60 CONTINUE
      CALL GENRAL
      TEST(1)=N
      IM=N
      IKEY=0
      DO 1 I=2,N
      IKEY=IKEY+1
      IM=TETA(IM)
      TEST(I)=IM
      L=TEST(I-1)
      LL=TEST(I)
      TEMP=CC(LL,L)
      CC(LL,L)=999999
      CC(L,LL)=-TEMP
      IF(TEST(I).EQ.IS) GO TO 2
    1 CONTINUE
    2 DO 20 I=1,N
      TEST(I)=0
      DO 20 J=1,N
   20 C(I,J)=CC(I,J)
      IF(KK.GT.5) GO TO 24
      DO 25 I=1,N
      WRITE(6,33)I
   25 WRITE(6,34)(C(I,II),II=1,N)
```

```
  34 FORMAT(1X,18I6)
  33 FORMAT(I4)
     KK=KK+1
  24 GO TO 5
 200 II=0
     COST=0
     DO 100 I=IL,N
     DO 100 J=1,N
     IF(CC(I,J).GE.0) GO TO 100
     COST=COST+CC(I,J)
     II=II+1
     LIST(II,1)=J
     LIST(II,2)=I
 100 CONTINUE
     WRITE(6,203)
 203 FORMAT(5X,'SCURCE',/)
     WRITE(6,101)(LIST(JJ,1),JJ=1,II)
     WRITE(6,204)
 204 FORMAT(5X,'SINK',/)
     WRITE(6,101)(LIST(JJ,2),JJ=1,II)
     COST=-COST
 101 FORMAT(8I4)
 102 FORMAT(4I2)
 104 FORMAT(5X,'TOTAL TRAVEL TIME=',I8)
     WRITE(6,104)COST
     STOP
     END
```

APÊNDICE B

Programa 2.a - Mētodo de Klein utilizando-se o Algorītmo de Flarian e Robert para detetar o circuito nega tivo.

```
      SUBROUTINE CYNEG
      INTEGER CYCLE(53),C(53,53),VALUE(53),OPEN(53),VAL,S,W,U
      COMMON N,C,KEY,CYCLE,L
      DO 19 I=1,N
      DO 2 J=1,N
      VALUE(J)=0
      CYCLE(J)=VALUE(J)
    2 OPEN(J)=1
      VALUE(1)=0
      OPEN(I)=VALUE(1)
      CYCLE(1)=I
      U=I
      L=1
    3 S=1
C *** SEARCH ALONG ROW U FROM COLUMN S TO COLUMN
C *** N AND FIND THE SMALLEST INDEX V            ***
    4 DO 8 JJ=S,N
      VAL=OPEN(JJ)*(VALUE(L)+C(U,JJ))
      IF(VAL.GE.0) GO TO 8
C *** ADD NEW POINT TO CYCLE                      ***
      OPEN(JJ)=0
      L=L+1
      CYCLE(L)=JJ
      VALUE(L)=VAL
      U=JJ
C ***  TEST WHETHER THE CYCLE MAY BE COMPLETED    ***
      W=CYCLE(1)
      IF(VALUE(L)+C(U,W))6,3,3
    6 CYCLE(L+1)=CYCLE(1)
      LL=L+1
      IF(LL.EQ.3) GO TO 21
      WRITE(6,23)(CYCLE(IP),IP=1,LL)
   23 FORMAT(14I6)
      KEY=1
      GO TO 20
    8 CONTINUE
    9 IF(L.EQ.1)GO TO 19
      S=CYCLE(L)
      VALUE(L)=0
      OPEN(S)=1
      IF(S.EQ.W)GO TO 15
      S=S+1
      L=L-1
      U=CYCLE(L)
      GO TO 4
   15 L=L-1
      GO TO 9
   19 CONTINUE
   20 RETURN
   21 KEY=0
      RETURN
      END
```

```
      INTEGER CYCLE(53),C(53,53),INFLOW(30,2),FVALUE,LIST(50,2),
     *CC(53,53),COST,TEMP
      COMMON N,C,KEY,CYCLE,L
      READ(5,102)IC,IL
      N=IC+1
      DO 3 I=1,N
      DO 3 J=1,N
      IF(I.EQ.J) GO TO 2
      C(I,J)=999999
      GO TO 3
    2 C(I,J)=0
    3 CONTINUE
      ILL=IL+1
      DO 24 I=2,IL
   24 READ(5,21)(C(I,II),II=ILL,IC)
   21 FORMAT(8I4)
      DO 4 I=1,IL
      C(1,I)=0
      C(IL+I,N)=0
    4 CONTINUE
      IS=1
      IT=N
      READ(5,33)FVALUE
      WRITE(6,105)
      DO 5 I=1,N
      WRITE(6,33)I
    5 WRITE(6,22)(C(I,J),J=1,N)
   22 FORMAT(1X,18I6)
      READ(5,23)(INFLOW(I,1),I=1,FVALUE)
      READ(5,23)(INFLOW(I,2),I=1,FVALUE)
   23 FORMAT(8I2)
      DO 10 J=1,FVALUE
      I=INFLOW(J,2)
      II=INFLOW(J,1)
      C(IS,II)=999999
      C(II,IS)=0
      TEMP=C(II,I)
      C(II,I)=C(I,II)
      C(I,II)=-TEMP
      C(I,IT)=999999
   10 C(IT,I)=0
      NN=N+1
      DO 26 I=1,N
      WRITE(6,33)I
      WRITE(6,22)(C(I,JJ),JJ=1,N)
      DO 26 J=1,N
   26 CC(I,J)=C(I,J)
    1 KEY=0
      CALL CYNEG
      IF(KEY.EQ.0) GO TO 20
      DO 40 I=1,L
      M=CYCLE(I)
      MM=CYCLE(I+1)
      TEMP=CC(M,MM)
      CC(M,MM)=999999
   40 CC(MM,M)=-TEMP
      DO 42 I=1,N
      CYCLE(I)=0
```

```
            DO 42 J=1,N
         42 C(I,J)=CC(I,J)
         33 FORMAT(I4)
            GO TO 1
         20 II=0
            WRITE(6,106)
            DO 201 I=1,N
            WRITE(6,33)I
        201 WRITE(6,22)(CC(I,JJ),JJ=1,N)
            COST=0
            DO 100 I=IL,N
            DO 100 J=1,N
            IF(CC(I,J).GE.0) GO TO 100
            COST=COST+CC(I,J)
            II=II+1
            LIST(II,1)=J
            LIST(II,2)=I
        100 CONTINUE
            WRITE(6,203)
            WRITE(6,101)(LIST(JJ,1),JJ=1,II)
            WRITE(6,204)
        203 FORMAT(5X,'SOURCE',/)
        204 FORMAT(5X,'SINK',/)
            WRITE(6,101)(LIST(JJ,2),JJ=1,II)
            COST=-COST
            WRITE(6,104)COST
        101 FORMAT(,25I4)
        102 FORMAT(2I2)
        104 FORMAT(5X,'TOTAL TRAVEL TIME=',I8)
        105 FORMAT(45X,'INITIAL MATRIX',/)
        106 FORMAT(/,45X,'FINAL MATRIX',/)
        200 STOP
            END
```

Programa 2.b - Método de Klein utilizando-se o algorítmo de Floyd para detetar o circuíto negativo.

```
      SUBROUTINE CYNEG(C,N,CYCLE,L,KEY)
      INTEGER*2 C(53,53),IETA(53,53),CYCLE(53)
      KEY=0
      DO 4 I=1,N
      DO 4 J=1,N
      IS=1
      IF(C(I,J).EQ.C(N,IS)) GO TO 3
      IETA(I,J)=I
      GO TO 4
    3 IETA(I,J)=C(N,IS)
    4 CONTINUE
      K=0
    8 K=K+1
      DO 40 I=1,N
      IF(I-K)10,40,10
   10 DO 30 J=1,N
      IF(J-K)15,30,15
   15 SUM=C(I,K)+C(K,J)
      IF(SUM-C(I,J))20,30,30
   20 C(I,J)=SUM
      IETA(I,J)=IETA(K,J)
   30 CONTINUE
      IF(C(I,I))50,40,40
   40 CONTINUE
      GO TO 70
C  ***  TERMINATIN TEST  ***
   50 WRITE(6,60)I
   60 FORMAT(5X,'A NEGATIVE COST CIRCUIT CONTAINRNG',I3,'EXISTS IN
     *GRAPH'/)
      KEY=1
      L=1
      CYCLE(L)=I
      M=IETA(I,I)
   51 L=L+1
      CYCLE(L)=M
      IF(M.EQ.I) GO TO 79
      M=IETA(I,M)
      GO TO 51
   79 WRITE(6,81)(CYCLE(I),I=1,L)
   81 FORMAT(20I5)
      GO TO 80
   70 IF(K-N)8,80,80
   80 RETURN
      END
```

```
      INTEGER*2 C(53,53),IETA(53,53),TEMP,CYCLE(53),INFLOW(25,2),FVALUE,
     *LIST(25,2),CC(53,53),COST
      IS=1
      READ(5,102)IC,IL,FVALUE
      N=IC+1
      DO 3 I=1,N
      DO 3 J=1,N
      IF(I.EQ.J) GO TO 2
      C(I,J)=999999
      GO TO 3
    2 C(I,J)=0
    3 CONTINUE
      ILL=IL+1
      DO 24 I=2,IL
   24 READ(5,21)(C(I,II),II=ILL,IC)
   21 FORMAT(8I4)
      DO 4 I=1,IL
      C(I,I)=0
      C(IL+I,N)=0
    4 CONTINUE
      IT=N
      WRITE(6,11)
   11 FORMAT(45X,'INITIAL MATRIX',/)
      DO 27 I=1,N
      WRITE(6,33)I
   27 WRITE(6,22)(C(I,II),II=1,N)
      READ(5,23)(INFLOW(I,1),I=1,FVALUE)
      READ(5,23)(INFLOW(I,2),I=1,FVALUE)
   23 FORMAT(8I2)
   22 FORMAT(1X,18I6)
      DO 10 J=1,FVALUE
      I=INFLOW(J,2)
      II=INFLOW(J,1)
      C(IS,II)=999999
      C(II,IS)=0
      TEMP=C(II,I)
      C(II,I)=C(I,II)
      C(I,II)=-TEMP
      C(I,IT)=999999
   10 C(IT,I)=0
      DO 26 I=1,N
      DO 26 J=1,N
   26 CC(I,J)=C(I,J)
    1 KEY=0
      CALL CYNEG(C,N,CYCLE,L,KEY)
      IF(KEY.EQ.0) GO TO 20
      DO 40 I=1,L
      IF(I.EQ.1) GO TO 35
      IF(CYCLE(I).EQ.CYCLE(1)) GO TO 41
   35 MM=CYCLE(I)
      M=CYCLE(I+1)
      TEMP=CC(M,MM)
      CC(M,MM)=CC(MM,M)
   40 CC(MM,M)=-TEMP
   41 DO 42 I=1,N
      CYCLE(I)=0
      DO 42 J=1,N
   42 C(I,J)=CC(I,J)
```

```
 33 FORMAT(I4)
    GO TO 1
 22 II=0
    DO 201 I=1,N
    WRITE(6,33)I
201 WRITE(6,22)(CC(I,JJ),JJ=1,N)
    COST=0
    DO 100 I=IL,N
    DO 100 J=1,N
    IF(CC(I,J).GE.0) GO TO 100
    COST=COST+CC(I,J)
    II=II+1
    LIST(II,1)=J
    LIST(II,2)=I
100 CONTINUE
    WRITE(6,101)(LIST(JJ,1),JJ=1,II)
    WRITE(6,103)(LIST(JJ,2),JJ=1,II)
    COST=-COST
    WRITE(6,104)COST
101 FORMAT(5X,'SOURCE',8I4)
102 FORMAT(3I2)
103 FORMAT(5X,'SINK',8I4)
104 FORMAT(5X,'TOTAL TRAVEL TIME=',I8)
    STOP
    END
```

APÊNDICE C

Programa 3 - Método de designação ótima. [14]

```
      INTEGER C(25,25),LL(50),CL(50),CZ(50),PI(50),PJ(50),KARTA(80),
     *COST,KEY1,LZ(50)
      COMMON N,C,LZ,CZ,PI,PJ,LL,CL,KEY,COST,LLL
      KEY2=0
      MM=0
    3 READ(5,1)N
      MM=MM+1
    1 FORMAT(I2)
      IF(N.EQ.0) GO TO 300
      COST=0
      DO 2 I=1,N
      LZ(I)=0
      CZ(I)=0
      PI(I)=0
      PJ(I)=PI(I)
      LL(I)=0
    2 CL(I)=LL(I)
      READ(5,205)KARTA
      WRITE(6,206)KARTA
      GO TO (1300,1300,1200,1200,1100),MM
 1100 DO 1101 I=1,N
 1101 READ(5,1001)(C(I,J),J=1,N)
 1001 FORMAT(6I4)
      GO TO 333
 1200 DO 1201 I=1,N
 1201 READ(5,1002)(C(I,J),J=1,N)
 1002 FORMAT(13I6)
      GO TO 333
 1300 DO 1301 I=1,N
 1301 READ(5,1003)(C(I,J),J=1,N)
 1003 FORMAT(8I4)
  333 WRITE(6,4)
    4 FORMAT(20X,'INITIAL MATRIX',/)
      IF(MM.GT.1) GO TO 216
      DO 209 I=1,N
      WRITE(6,210)I
  209 WRITE(6,208)(C(I,J),J=1,N)
  208 FORMAT(8I6)
  210 FORMAT(I4)
      GO TO 211
  216 WRITE(6,207)((C(I,J),J=1,N),I=1,N)
  207 FORMAT(6I8)
  211 DO 8 I=1,N
      MIN=C(I,1)
      DO 6 J=1,N
      IF(C(I,J)-MIN) 5,6,6
    5 MIN=C(I,J)
    6 CONTINUE
      IF(MIN.EQ.0) GO TO 8
      COST=COST+MIN
      DO 7 IP=1,N
    7 C(I,IP)=C(I,IP)-MIN
    8 CONTINUE
      DO 13 I=1,N
      MIN=C(1,I)
      DO 10 J=1,N
      IF(C(J,I)-MIN) 9,10,10
    9 MIN=C(J,I)
```

```
      10 CONTINUE
         IF(MIN)11,13,11
      11 COST=COST+MIN
         DO 12 ID=1,N
      12 C(ID,I)=C(ID,I)-MIN
      13 CONTINUE
      14 KEY=0
         DO 23 I=1,N
         DO 22 J=1,N
         IF(C(I,J))22,15,22
      15 LZ(I)=LZ(I)+1
         CZ(J)=CZ(J)+1
         IF(LZ(I)-1)16,16,17
      16 PJ(I)=J
         LL(I)=1
         GO TO 18
      17 LL(I)=0
         PJ(I)=0
      18 IF(CZ(J).GT.1) GO TO 20
         PI(J)=I
         CL(J)=1
         GO TO 22
      20 CL(J)=0
         PI(J)=0
      22 CONTINUE
      23 CONTINUE
      25 DO 30 I=1,N
C     *** FINAL ASSIGNMENT IN EACH LINE  ***
         IF(LL(I).NE.1) GO TO 30
         IF(LZ(I).EQ.0) GO TO 30
         JJ=PJ(I)
         CALL DESIN(I,JJ)
         CALL ELMIN(I,JJ)
      30 CONTINUE
         DO 40 K=1,N
         IF(CL(K).NE.1) GO TO 40
         IF(CZ(K).EQ.0) GO TO 40
         II=PI(K)
         CALL DESIN(II,K)
         CALL ELMIN(II,K)
      40 CONTINUE
         DO 45 I=1,N
         IF(CZ(I).EQ.1) GO TO 25
      45 CONTINUE
         DO 50 I=1,N
         IF(LZ(I).EQ.1) GO TO 25
      50 CONTINUE
         IF(KEY.LT.N) GO TO 80
         WRITE(6,60)
      60 FORMAT(//5X,12HFINAL MATRIX,/)
         IF(MY.GT.1) GO TO 62
         DO 61 I=1,N
         WRITE(6,210)I
      61 WRITE(6,208)(C(I,J),J=1,N)
         GO TO 66
      62 WRITE(6,207)((C(I,J),J=1,N),I=1,N)
      66 WRITE(6,70)(J,PI(J),J=1,N)
         WRITE(6,75)COST
```

```
   70 FORMAT(I10,25I4)
   75 FORMAT(/,5X,17HTOTAL TRAVEL TIME,I8/)
      GO TO 3
   80 DO 95 I=1,N
      IF(LZ(I).EQ.0) GO TO 95
      DO 85 J=1,N
      IF(CL(J).NE.0) GO TO 85
      IF(C(I,J).NE.0) GO TO 85
      KK=J
      GO TO 90
   85 CONTINUE
   90 CALL DESIN(I,KK)
      CALL ELMIN(I,KK)
      GO TO 25
   95 CONTINUE
      DO 98 I=1,N
      IF(LL(I).NE.0) GO TO 98
      LZ(I)=1
   98 CONTINUE
   96 KEY1=0
      DO 100 I=1,N
      IF(LZ(I).NE.1) GO TO 100
      DO 97 J=1,N
      IF(C(I,J).NE.0) GO TO 97
      IF(CZ(J).EQ.1) GO TO 97
      CZ(J)=1
      KEY1=1
   97 CONTINUE
  100 CONTINUE
      DO 110 I=1,N
      IF(CZ(I).NE.1) GO TO 110
      IF(CL(I).EQ.0) GO TO 110
      II=PI(I)
      IF(LZ(II).EQ.1) GO TO 110
      KEY1=1
      LZ(II)=1
  110 CONTINUE
      IF(KEY1.NE.0) GO TO 96
      MIN=999999
      DO 120 J=1,N
      IF(LZ(J).EQ.0) GO TO 120
      DO 115 I=1,N
      IF(CZ(I).EQ.1) GO TO 115
      IF(C(J,I).GT.MIN) GO TO 115
      MIN=C(J,I)
  115 CONTINUE
  120 CONTINUE
      DO 130 I=1,N
      COST=COST+MIN
      DO 130 J=1,N
  130 C(I,J)=C(I,J)-MIN
      DO 160 I=1,N
      IF(LZ(I).EQ.1) GO TO 160
      COST=COST-MIN
      DO 155 J=1,N
  155 C(I,J)=C(I,J)+MIN
  160 CONTINUE
      DO 170 J=1,N
```

```
      IF(CZ(J).EQ.0) GC TO 170
      COST=COST-MIN
      DO 165 I=1,N
  165 C(I,J)=C(I,J)+MIN
  170 CONTINUE
      KEY2=KEY2+1
      DO 200 I=1,N
      CZ(I)=0
  200 LZ(I)=0
      GO TC 14
  205 FORMAT(80A1)
  206 FORMAT(1H1,/,1X,80A1)
  300 STCP
      END
```

```
0001      SUBROUTINE DESIN(I,J)
0002      INTEGER C(25,25),LL(50),CL(50),COST,LZ(50),CZ(50),PI(50),PJ(50),
         *M(50)
0003      COMMON N,C,LZ,CZ,PI,PJ,LL,CL,KEY,COST,LLL
0004      LZ(I)=0
0005      CZ(J)=0
0006      CL(J)=1
0007      PI(J)=I
0008      KEY=KEY+1
0009      LL(I)=1
0010      PJ(I)=J
0011      RETURN
0012      END
```

FORTRAN IV G LEVEL 21 ELMIN DATE = 77209 17/04/44 PAGE 0001

```
      SUBROUTINE ELMIN(K,L)
      INTEGER C(25,25),LL(50),CL(50),COST,LZ(50),CZ(50),PI(50),PJ(50),
     *JJ(50)
      COMMON N,C,LZ,CZ,PI,PJ,LL,CL,KEY,COST,LLL
      DO 10 I=1,N
      IF(LZ(I).EQ.0) GO TO 10
      IF(C(I,L).NE.0) GO TO 10
      LZ(I)=LZ(I)-1
      IF(LZ(I).GT.0) GO TO 5
      LL(I)=0
      PJ(I)=LL(I)
      GO TO 10
    5 IF(LZ(I).NE.1) GO TO 10
      LL(I)=1
      DO 6 J=1,N
      IF(C(I,J).NE.0) GO TO 6
      IF(CZ(J).LE.0) GO TO 6
      PJ(I)=J
    6 CONTINUE
   10 CONTINUE
      DO 30 J=1,N
      IF(CZ(J).EQ.0) GO TO 30
      IF(C(K,J).NE.0) GO TO 30
      CZ(J)=CZ(J)-1
      IF(CZ(J).GT.0) GO TO 20
      CL(J)=0
      PI(J)=0
      GO TO 30
   20 IF(CZ(J).NE.1) GO TO 30
      CL(J)=1
      DO 25 I=1,N
      IF(C(I,J).NE.0) GO TO 25
      IF(LZ(I).LE.0) GO TO 25
      PI(J)=I
   25 CONTINUE
   30 CONTINUE
      RETURN
      END
```

APÊNDICE D

Exemplo 1.a - Resultados obtidos dos programas 1 e 2 para primeira região.

INITIAL MATRIX

```
 1
     0     0     0     0     0     0     0     0     0999999999999999999999999999999999999999999999999999999
 2
999999     0999999999999999999999999999999999999999999   331  1056  1130  1155  1780  1780  1862  1862999999
 3
999999999999     0999999999999999999999999999999999999   675   520  1087  1021  1646  1646  1545  1545999999
 4
999999999999999999     0999999999999999999999999999999  1858  1083   725   629   346   346   108   108999999
 5
999999999999999999999999     0999999999999999999999999  1858  1083   725   629   346   346   108   108999999
 6
999999999999999999999999999999     0999999999999999999  1858  1083   725   629   346   346   108   108999999
 7
999999999999999999999999999999999999     0999999999999  1858  1083   725   629   346   346   108   108999999
 8
999999999999999999999999999999999999999999     0999999  1858  1083   725   629   346   346   108   108999999
 9
999999999999999999999999999999999999999999999999     0  1858  1083   725   629   346   346   108   108999999
 10
999999999999999999999999999999999999999999999999999999     0999999999999999999999999999999999999999999     0
 11
999999999999999999999999999999999999999999999999999999999999     0999999999999999999999999999999999999     0
 12
999999999999999999999999999999999999999999999999999999999999999999     0999999999999999999999999999999     0
 13
999999999999999999999999999999999999999999999999999999999999999999999999     0999999999999999999999999     0
 14
999999999999999999999999999999999999999999999999999999999999999999999999999999     0999999999999999999     0
 15
999999999999999999999999999999999999999999999999999999999999999999999999999999999999     0999999999999     0
 16
999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999     0999999     0
 17
999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999     0     0
 18
999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999     0
 1
     0999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999
 2
     0     0999999999999999999999999999999999999999999999999  1056  1130  1155  1780  1780  1862  1862999999
 3
     0999999     0999999999999999999999999999999999999   675   520  1087  1021  1646  1646999999  1545999999
 4
     0999999999999     0999999999999999999999999999999  1858  1083   725   629   346999999   108   108999999
 5
     0999999999999999999     0999999999999999999999999  1858  1083999999   629   346   346   108   108999999
 6
     0999999999999999999999999     0999999999999999999  1858  1083   725999999   346   346   108   108999999
 7
     0999999999999999999999999999999     0999999999999  1858  1083   725   629999999   346   108   108999999
 8
     0999999999999999999999999999999999999     0999999  1858999999   725   629   346   346   108   108999999
 9
     0999999999999999999999999999999999999999999     0  1858  1083   725   629   346   346   108999999999999
 10
999999  -331999999999999999999999999999999999999999999     0999999999999999999999999999999999999999999999999
 11
999999999999999999999999999999999999999999 -1083999999999999     0999999999999999999999999999999999999999999
```

```
12
999999999999999999999999999  -72599999999999999999999999999999999999999999      0999999999999999999999999999999999999
13
99999999999999999999999999999999  -62999999999999999999999999999999999999999999      09999999999999999999999999999999999
14
9999999999999999999999999999999999999  -3469999999999999999999999999999999999999999      099999999999999999999999999999
15
9999999999999999999  -34699999999999999999999999999999999999999999999999999999999999999      09999999999999999999999
16
999999999999 -15459999999999999999999999999999999999999999999999999999999999999999999999999      09999999999999
17
999999999999999999999999999999999999999999999999999999  -10399999999999999999999999999999999999999999999999      0999999
18
999999999999999999999999999999999999999999999999999999999999999      0      0      0      0      0      0      0      0      0
   11    8    12    5    13    6    14    7    15    4    16    3    11

                              FINAL MATRIX

 1
   0999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999
 2
   0      09999999999999999999999999999999999999999999999999999  1056  1130  1155  1780  1780  1862  1862999999
 3
   0999999      09999999999999999999999999999999999999999999  675999999  1037  1021  1646  1646  1545  1545999999
 4
   09999999999999      0999999999999999999999999999999999999  1858  1083   725   629   346   346999999   108999999
 5
   09999999999999999999      09999999999999999999999999999999  1858  1083   725999999   346   346   108   108999999
 6
   0999999999999999999999999      0999999999999999999999999  1858  1083   725   629999999   346   108   108999999
 7
   0999999999999999999999999999999999      0999999999999999  1858  1083   725   629   346999999   108   108999999
 8
   09999999999999999999999999999999999999999      0999999  1858  1083999999   629   346   346   108   108999999
 9
   0999999999999999999999999999999999999999999999      0  1858  1083   725   629   346   346   108999999999999
10
999999  -33199999999999999999999999999999999999999999999999      09999999999999999999999999999999999999999999999999999
11
99999999999  -52099999999999999999999999999999999999999999999999      099999999999999999999999999999999999999999999999
12
9999999999999999999999999999999999999999999999999999999  -72599999999999999999999      0999999999999999999999999999999999999999
13
999999999999999999999999999  -62999999999999999999999999999999999999999999999999      09999999999999999999999999999999999
14
99999999999999999999999999999999999  -1469999999999999999999999999999999999999999999999      099999999999999999999999999
15
9999999999999999999999999999999999999999999  -3469999999999999999999999999999999999999999999999999      09999999999999999999
16
999999999999999999999  -108999999999999999999999999999999999999999999999999999999999999999999999999      09999999999999
17
999999999999999999999999999999999999999999999999999999999999  -1089999999999999999999999999999999999999999999999999      0999999
18
9999999999999999999999999999999999999999999999999999999999999999      0      0      0      0      0      0      0      0      0
   SOURCE

 2   3   8   5   6   7   4   9
   SINK

10  11  12  13  14  15  16  17
   TOTAL TRAVEL TIME=      3113
```

Exemplo 1.b - Resultados obtidos do programa 3 para primeira região.

***FIRST REGION ***

INITIAL MATRIX

1	331	1056	1130	1155	1780	1780	1862	1862
2	679	520	1037	1021	1646	1646	1545	1545
3	1858	1083	725	629	346	346	108	108
4	1858	1083	725	629	346	346	108	108
5	1858	1083	725	629	346	346	108	108
6	1858	1083	725	629	346	346	108	108
7	1858	1083	725	629	346	346	108	108
8	1858	1083	725	629	346	346	108	108

FINAL MATRIX

1	0	725	232	353	1261	1261	1581	1581
2	155	0	0	30	938	938	1075	1075
3	1700	925	0	0	0	0	0	0
4	1700	925	0	0	0	0	0	0
5	1700	925	0	0	0	0	0	0
6	1700	925	0	0	0	0	0	0
7	1700	925	0	0	0	0	0	0
8	1700	925	0	0	0	0	0	0

1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8

TOTAL TRAVEL TIME 3113