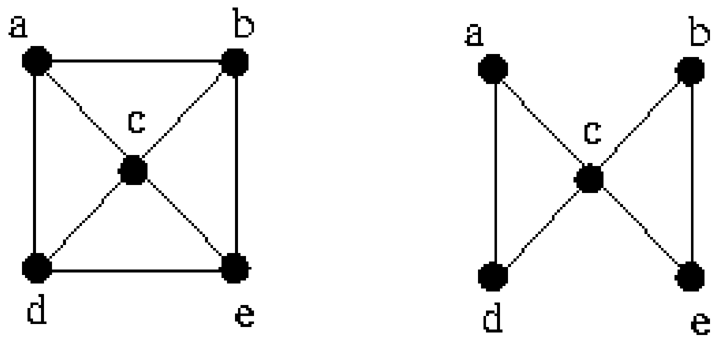
Grafos Hamiltonianos e o Problema do Caixeiro Viajante

Prof. Ademir Constantino
Departamento de Informática
Universidade Estadual de Maringá

Grafo Hamiltoniano

- **Definição**: Um **circuito hamiltoniano** em um grafo conexo *G* é definido como um caminho simples, fechado passando em cada vértice de *G* exatamente uma vez. Um grafo que admite um circuito hamiltoniano é um **grafo hamiltoniano**.



(a) (b)

Propriedades

- **Teorema de Ore**. Uma condição suficiente (mas não necessária) para que um grafo G seja hamiltoniano é que a soma dos graus de cada par de vértices não adjacentes seja no mínimo n.

- **Teorema de Dirac**: Uma condição suficiente (mas não necessária) para que um grafo simples G possua um ciclo hamiltoniano, é que o grau de cada vértice em G seja pelo menos igual a n/2, onde n é o número de vértices em G.

Algoritmos

- Não se conhece algoritmo exato de complexidade polinomial para encontrar caminhos hamiltonianos.

Prob. do Caixeiro Viajante

Dado um grafo $G=(V,E)$ conexo com pesos nas arestas, o objetivo do ***Problema do Caixeiro Viajante*** é encontrar um caminho fechado de peso mínimo passando por cada vértices pelo menos uma vez.

Algoritmos para o PCV

- Por que o PCC pode ser resolvido em tempo polinomial e o PCV não?

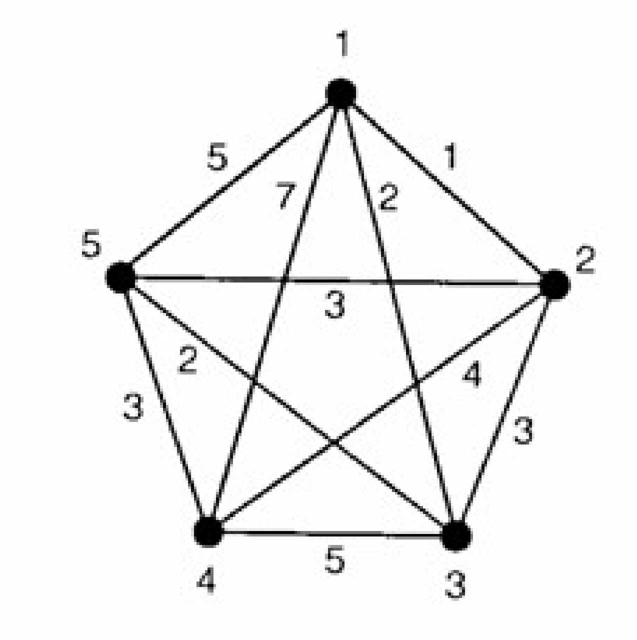
- Alternativa para obtenção de solução viável para o PCV.
  - Algoritmos heurísticos
    - Não há provas que o este tipo de algoritmo obtem a solução ótima para o problema, porém, são algoritmos de complexidade polinomial e relativamente fáceis de implementar.

Heurísticas para o PCV

- Construtivas
  - Vizinho mais próximo;
  - Inserção mais próxima;
  - Inserção mais distante;
  - Inserção mais barata;
  - Algoritmos das Economias (Clark-Wright).
- Melhorativas
  - K-opt ou K-melhoramento;
  - *Simulated Annealing;*
  - Algoritmos Genéticos;
  - Busca Tabu.

Exemplo

- Considere a seguinte matriz de custo para o grafo:

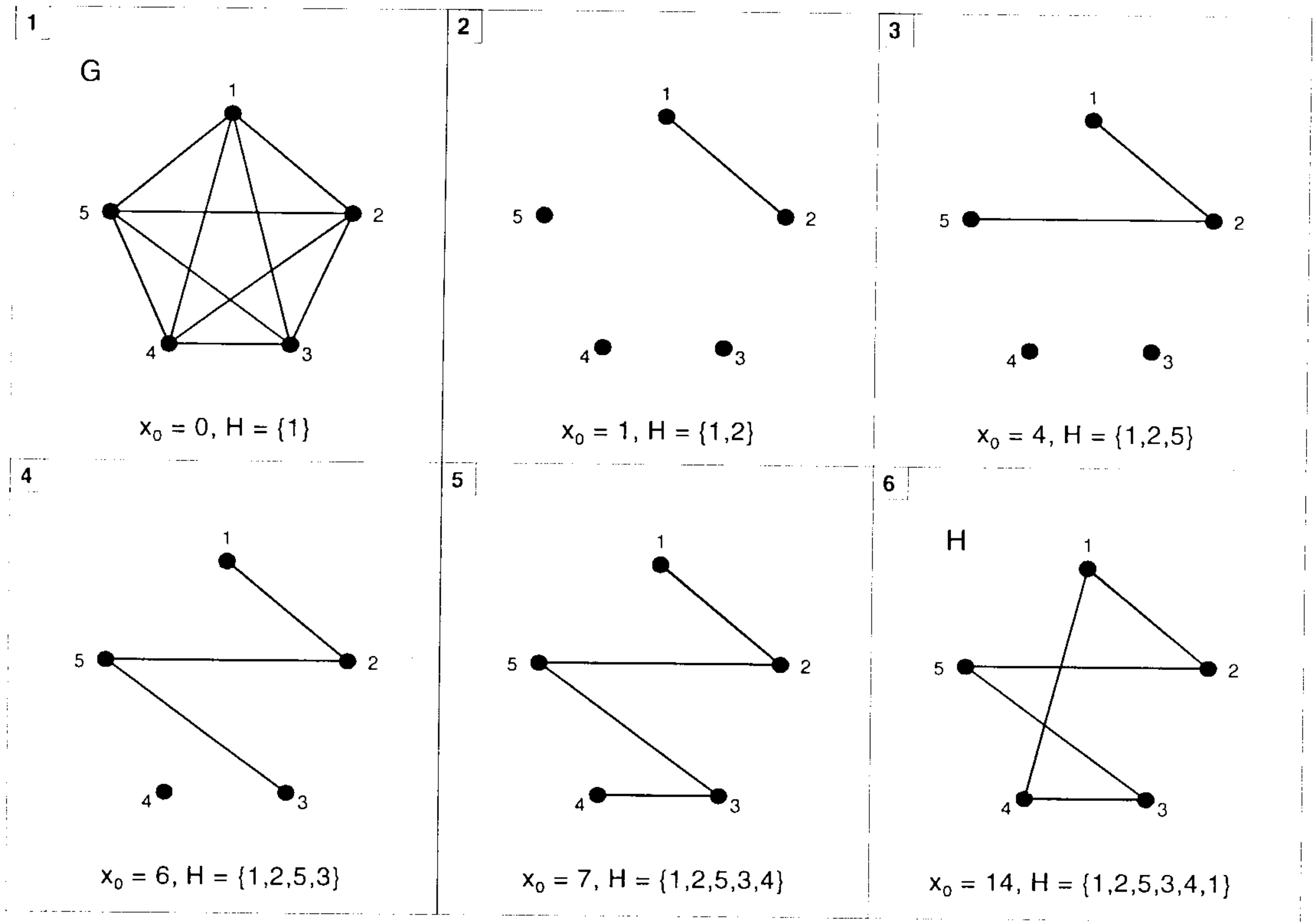


$$C = \begin{vmatrix} 0 & 1 & 2 & 7 & 5 \\ 1 & 0 & 3 & 4 & 3 \\ 2 & 3 & 0 & 5 & 2 \\ 7 & 4 & 5 & 0 & 3 \\ 5 & 3 & 2 & 3 & 0 \end{vmatrix}$$

Vizinho Mais Próximo

a) Iniciar com um vértice *v* qualquer e inicie um roteiro.
b) Escolher um vértice mais próximo do último vértice inserido no roteiro.
c) Se todos os vértices já foram inseridos, pare, caso contrário, volte ao passo “b”.

Vizinho Mais Próximo



ig. 6.8 *Tour* gerado pela Heurística VP

Inserção do Mais Próxima

Iniciar com um ciclo [v_1, v_2, v_3] com 3 vértices.

a) Encontrar um vértice v_k não pertencente ao ciclo, **mais próximo** de qualquer vértice do ciclo.

b) Encontrar uma aresta, digamos (v_i, v_{i+1}) do ciclo tal que:

$$(c_{i,k}+c_{k,i+1}-c_{i,i+1})$$

seja mínimo.

c) Inserir o vértice v_k entre (v_i, v_{i+1}). Se todos os vértices já foram inseridos, pare, caso contrário, voltar ao passo “b”.

Inserção do Mais Próxima

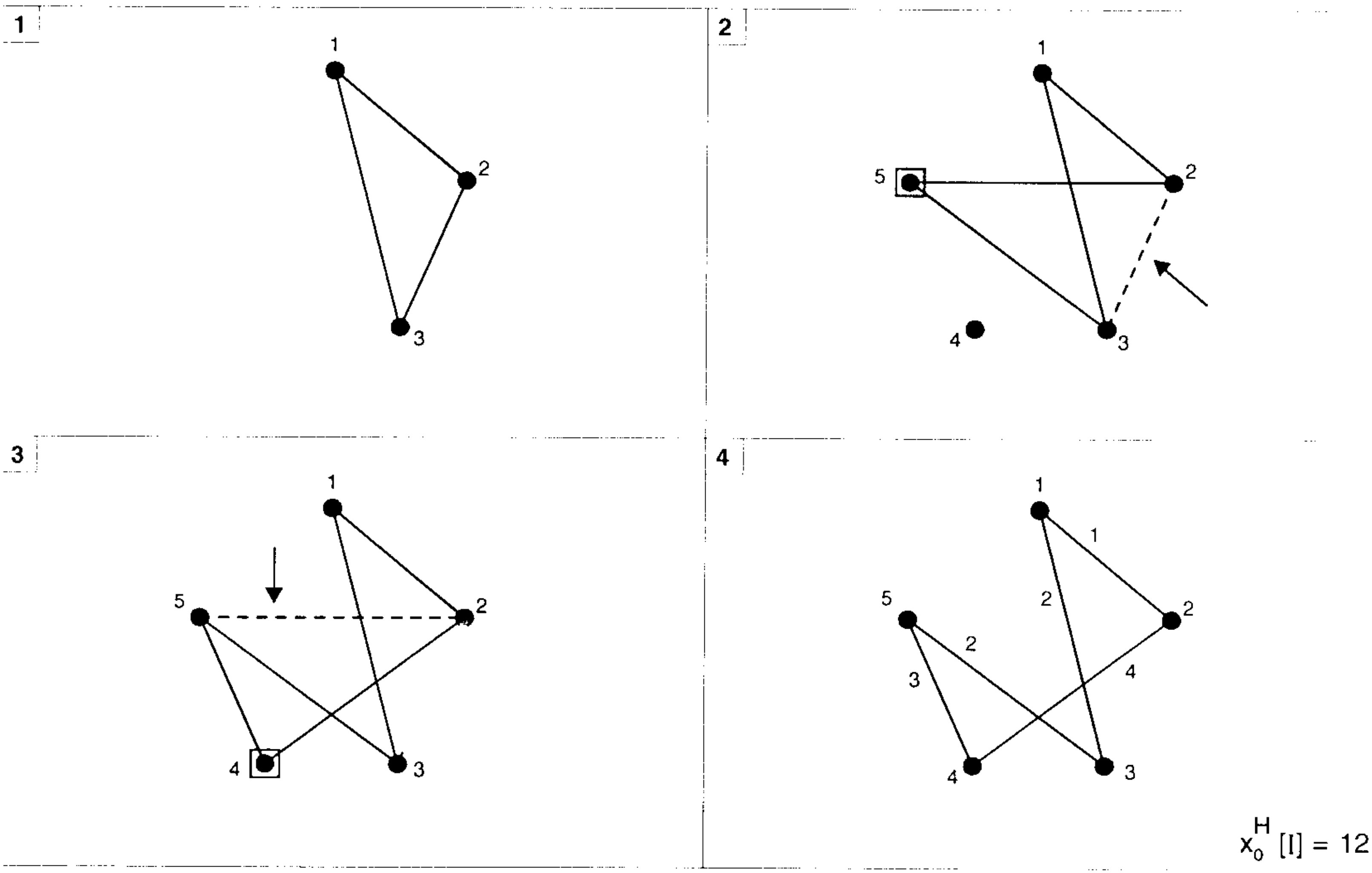
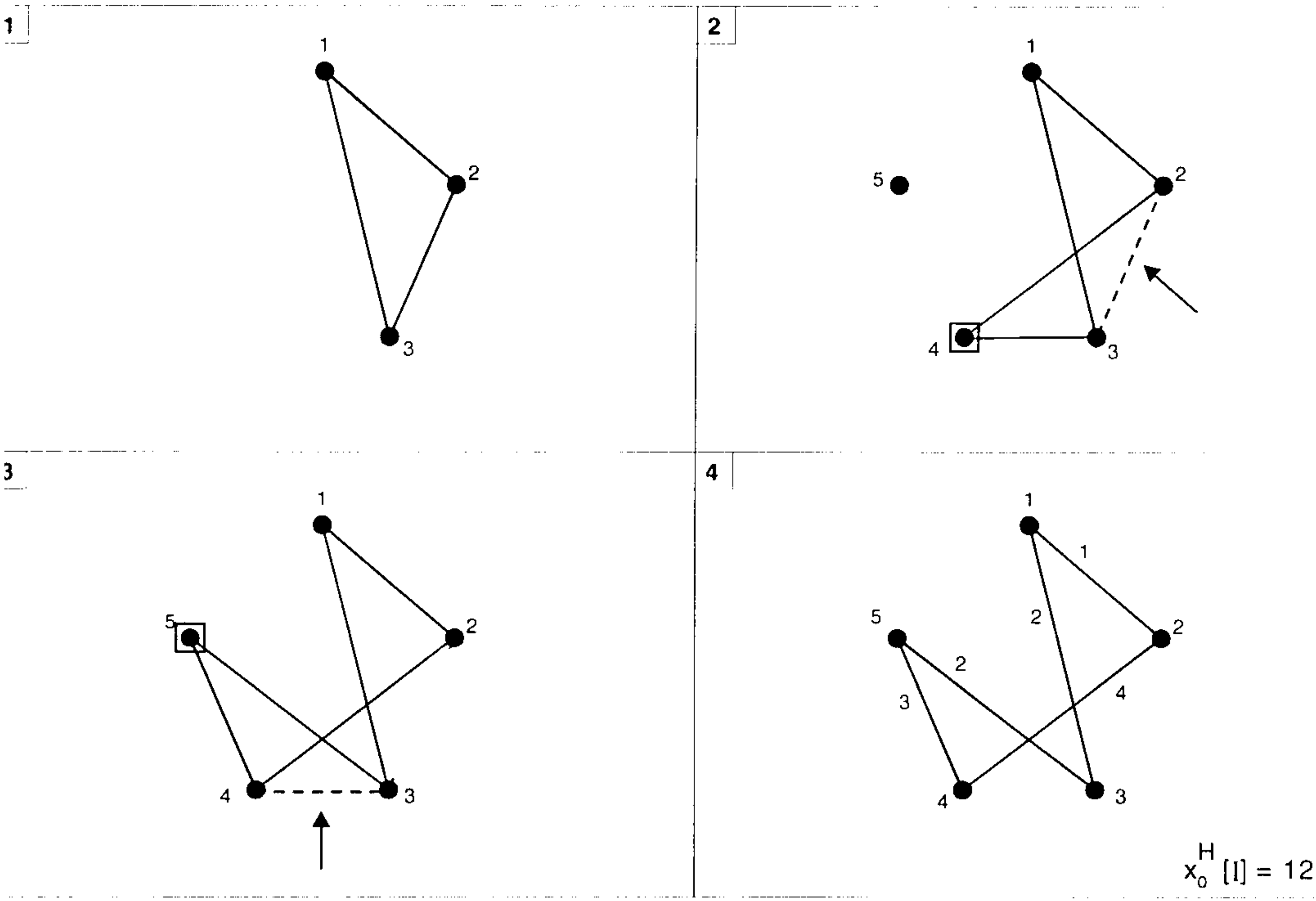


Fig. 6.10 Inserção Mais Próxima e Mais Barata

Inserção Mais Distante

- Difere do algoritmo anterior por, alínea “b”, escolher o vértice **mais distante**.

Inserção Mais Distante



ig. 6.11 Inserção Mais Distante

Inserção Mais Barata

Os passos "b" e "c" do algoritmo de Inserção Mais Próxima são substituidos por: encontrar um vértice v_k não pertencente ao ciclo e uma aresta do ciclo, digamos (v_i, v_{i+1}), tal que:

$$(c_{i,k} + c_{k,i+1} - c_{i,i+1})$$

seja **mínimo**.

Inserção Mais Barata

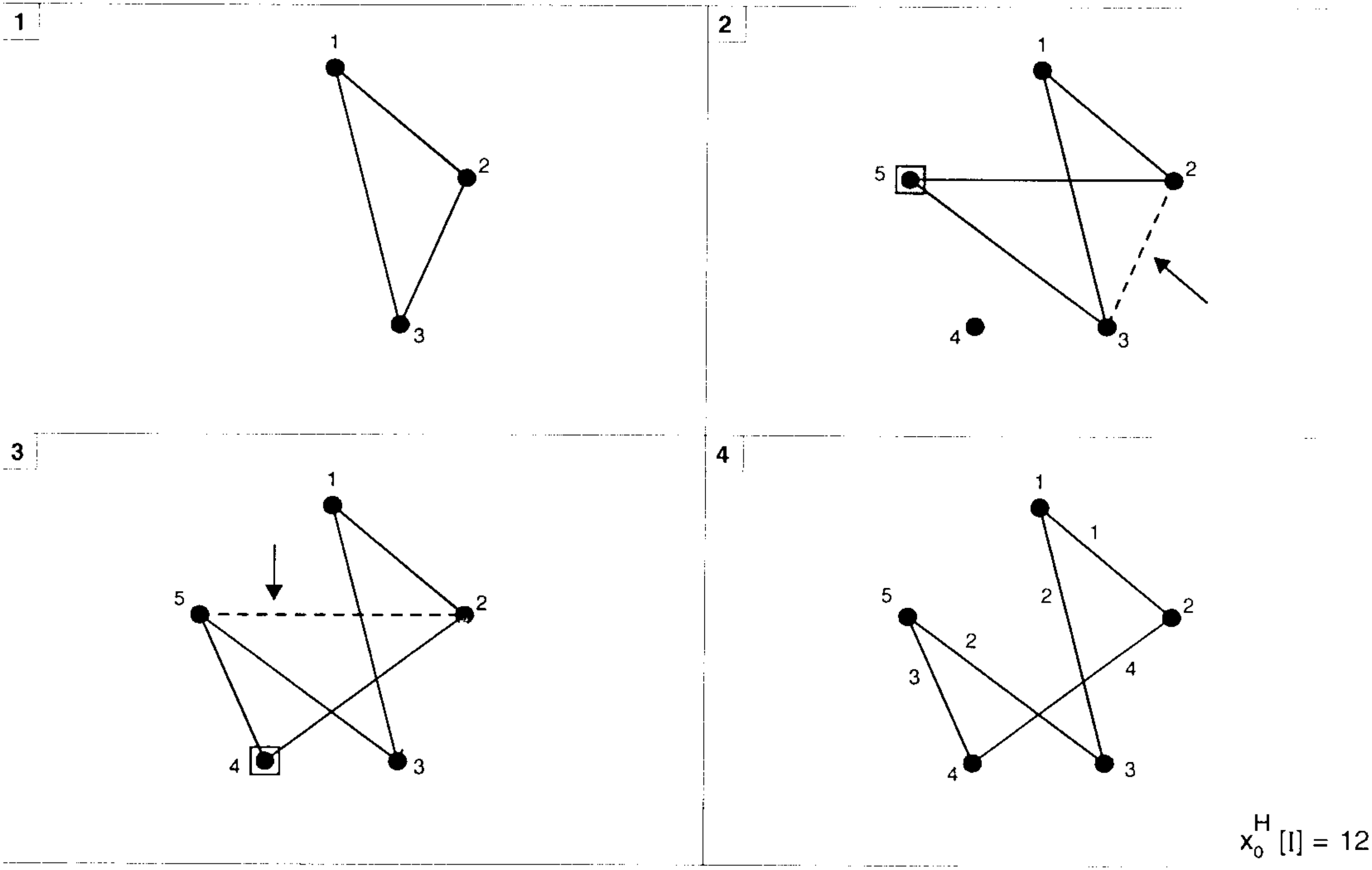


Fig. 6.10 Inserção Mais Próxima e Mais Barata

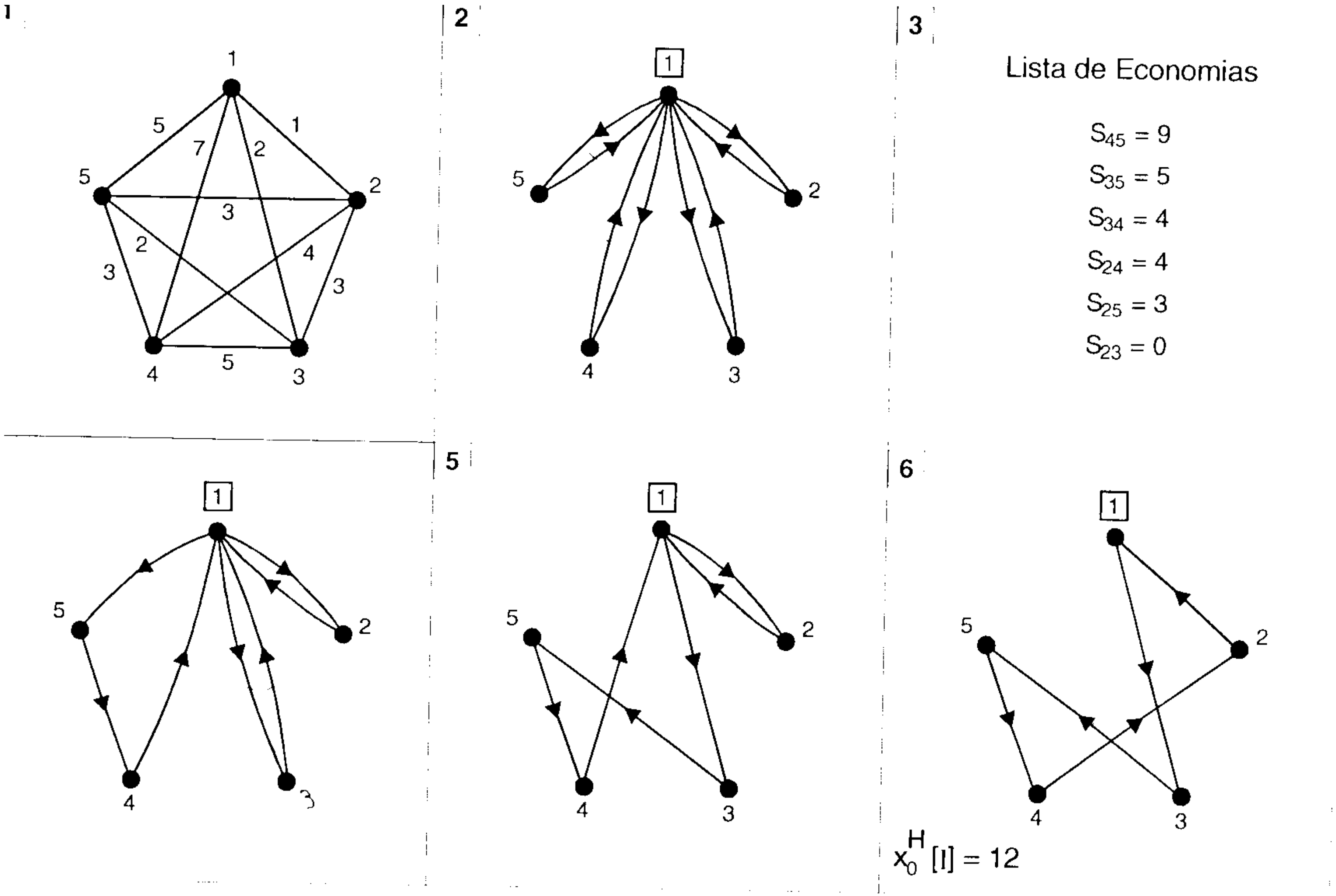
Algoritmos das Economias (Clark-Wright).

Passo 1: Calcule as economias $s_{i,j} = c_{v0,\ i} + c_{v0,\ j} - c_{i,\ j}$ para todos os pares de vértices (i, j), v_0 o vértice escolhido como inicial.

Passo 2: Ordene as enconomias $s_{i,j}$ em ordem não crescente (lista de economias).

Passo 3: Percorrer sequencialmente a lista de economias, inciando com a primeira. Tentar a ligação correspondente do primeiro par de vértices (i, j) da lista. Se a inserção da aresta (i, j) resultar num novo ciclo iniciando em v0, então eliminar $s_{i,j}$. Caso contrário, tentar a ligação do próximo da lista. Repetir até atingir o fim da lista.

Algoritmos das Economias (Clark-Wright).



6.15 Métodos das Economias

Algoritmos das Economias (Clark-Wright).

Complexidade final: $O(n^2 \log n)$

Porém, se o procedimento for repetido tomando cada vértice como inicial, então a complexidade passa para $O(n^3 \log n)$

Melhoria: k-Opt

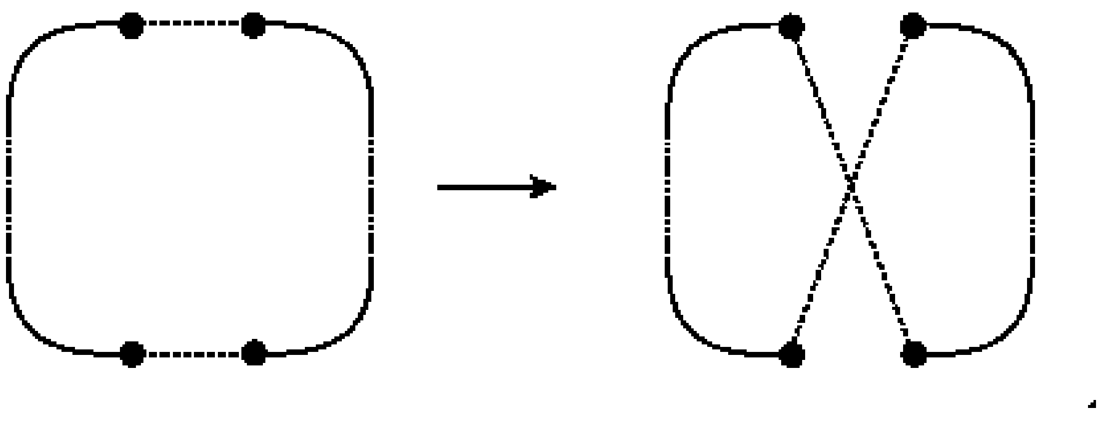
Seja H o ciclo encontrado por um algoritmo contrutivo.

Passos do Algoritmo:

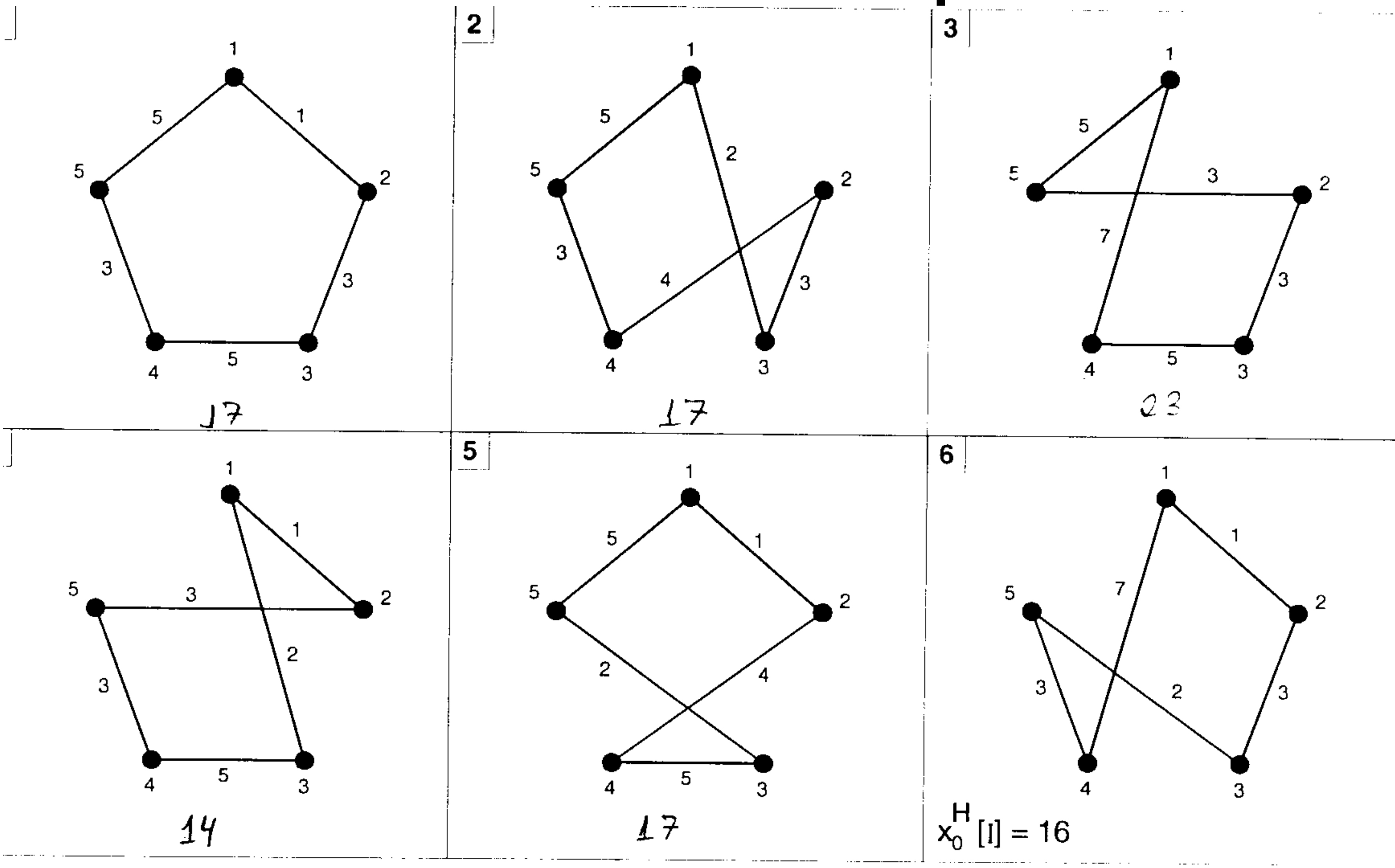
a) Remover k arestas da solução H obtendo uma solução H'.
b) Construir todas as soluções viáveis contendo H'.
c) Escolher a melhor soluções dentre as encontradas e guardar.
d) Escolher outro conjunto de k arestas ainda não selecionado e retornar ao passo "a", caso contrário, pare.

Melhoria: 2-Opt

Exemplo: removendo duas arestas, surge apenas uma possibilidade de recominações



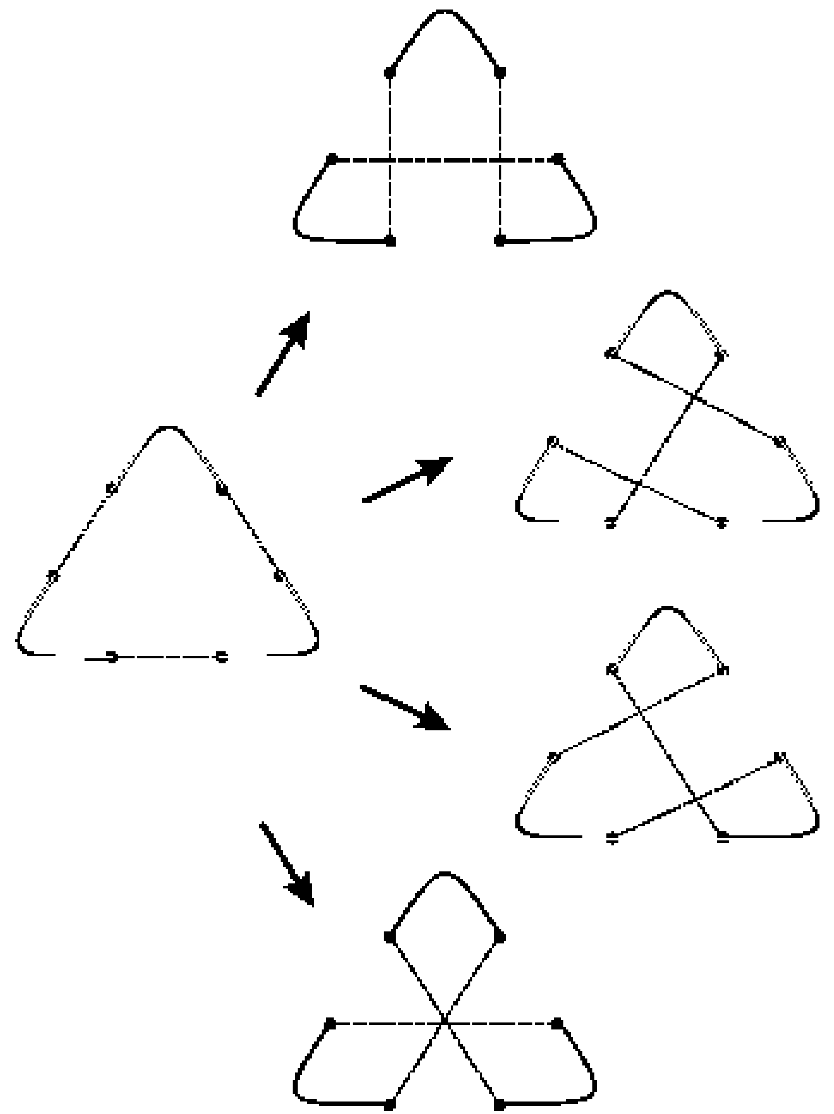
Melhoria: 2-Opt



]. 6.14 Heurística 2-Opt

Melhoria: 3-Opt

Exemplo: removendo três arestas, surgem quarto possibilidades de recominações



Conclusões

- As experiências tem mostrado que a heurística de Inserção Mais Distante ofecere um excelente resultado.

- Em geral quanto maior o valor de K maiores serão as chances de se obter a solução ótima com o procedimento K-Opt. Entretanto, o número de operações cresse rapidamente.

- As experiências computacionais tem mostrado que K=2 e K=3 oferecem excelentes resultados.