UNIVERSIDADE FEDERAL DO AMAZONAS - UFAM
FACULDADE DE TECNOLOGIA – FT
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

OTIMIZAÇÃO DA SEQÜÊNCIA DE MONTAGEM DE COMPONENTES PARA PROGRAMAÇÃO DE INSERSORAS AUTOMÁTICAS USANDO MÉDOTOS HEURÍSTICOS LOCAIS

ÁLVARO LUIZ MENEZES DE OLIVEIRA

MANAUS - 2008

OTIMIZAÇÃO DA SEQÜÊNCIA DE MONTAGEM DE COMPONENTES PARA PROGRAMAÇÃO DE INSERSORAS AUTOMÁTICAS USANDO MÉDOTOS DE BUSCA LOCAIS

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Amazonas, como requisito parcial para a obtenção do título de Mestre em Engenharia Elétrica.

Prof. Dr. Cícero F. F. Costa Filho - Orientador.
Prof. Dra. Marly Guimarães Fernandes Costa – Co-orientadora.

MANAUS – 2008

Ficha Catalográfica
(Catalogação realizada pela Biblioteca Central da UFAM)

Oliveira, Álvaro Luiz Menezes de

O48o Otimização da seqüência de montagem de componentes para programação de insersoras automáticas usando métodos heurísticos locais / Álvaro Luiz Menezes de Oliveira. - Manaus: UFAM, 2008.
134 f.; il.

Dissertação (Mestrado em Engenharia Elétrica) — Universidade Federal do Amazonas, 2008.
Orientador: Prof. Dr. Cícero F. F. Costa Filho
Co-orientadora: Profª. Dra.. Marly Guimarães Fernandes Costa

1. Máquinas de inserção automática 2. Componentes eletrônicos I. Costa Filho, Cícero F. F. II. Costa, Marly Guimarães Fernandes III. Universidade Federal do Amazonas IV. Título

CDU 62-52(043.3)

ÁLVARO LUIZ MENEZES DE OLIVEIRA

OTIMIZAÇÃO DA SEQÜÊNCIA DE MONTAGEM DE COMPONENTES PARA PROGRAMAÇÃO DE INSERSORAS AUTOMÁTICAS USANDO MÉDOTOS HEURÍSTICOS LOCAIS

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Amazonas, como requisito parcial para a obtenção do título de Mestre em Engenharia Elétrica, área de concentração Controle e Automação de Sistemas.

Aprovada em 09 de dezembro de 2008.

BANCA EXAMINADORA

Cícero Ferreira Fernandes Costa Filho, Dr. Eng
Presidente (UFAM)

Amit Bhaya, PhD
Membro Externo. (UFRJ)

Ramon Romankevicius Costa, Dr. Eng
Membro (UFRJ)

DEZEMBRO DE 2008

A minha inesquecível avó Cecília que sempre me dedicou seu mais puro carinho.

A meus pais, esposa e dois filhos que sempre me apoiaram em tudo.

Agradecimentos

Ao Deus onipotente e onisciente sem o qual nada é possível.

Aos meus pais, Álvaro e Terezinha, que sempre me educaram e apoiaram e sem os quais não seria possível a minha participação neste programa de mestrado.

A minha esposa Rosana que sempre demonstrou compreensão e apoio ao longo deste período; E também aos meus pequeninos filhos, Luiza e Luiz, que sempre souberam esperar a hora de brincar.

Ao professor Cícero Costa por sua inestimável contribuição e orientação na elaboração de todas as etapas deste trabalho e por sempre ter mostrado o lado positivo nas adversidades.

A professora Marly Guimarães por sua contribuição e sempre ter atendido as minhas solicitações.

Ao Centro de P&D em Tecnologia Eletrônica e da Informação – CETELI, por seu apoio em infra-estrutura cedida. Bem como a Faculdade de Tecnologia da UFAM pela oportunidade concedida através da realização deste programa de mestrado em engenharia elétrica.

A todos os professores que contribuíram para minha formação profissional, tanto da graduação quanto do curso de mestrado, em especial aos professores Edgar Filho e Valdir Sampaio, por terem me incentivado a participar do curso.

Também agradeço ao apoio da Sony Brasil e toda sua equipe técnica, em especial aos senhores Carlos Fonseca e Elidelson Carvalho, por terem concedido inestimáveis informações de seus trabalhos de pesquisa. Da mesma forma aos técnicos de IA, César Dantas, Thiago Escossio, Virgínia Miranda, por sua paciência e apoio técnico na operação das máquinas de inserção automática da Sony.

Resumo

Neste trabalho abordou-se o problema de otimização da seqüência de montagem de componentes eletrônicos sobre as placas de circuito impresso, PCI. A solução para este problema equivaleria a encontrar a melhor seqüência de montagem de todos os componentes da placa que garantisse uma melhor produtividade desta atividade industrial. A solução também precisaria ser encontrada no menor espaço de tempo possível, uma vez que isto seria de grande utilidade na avaliação de diversas seqüências simuladas. Trata-se do desejo fundamental da indústria de montagem automática de componentes, as quais utilizam máquinas de alta velocidade e algoritmos especializados no sentido de alcançar a máxima produtividade.

Muitos trabalhos já foram desenvolvidos com este objetivo, e progressivamente alcançaram resultados vantajosos em relação a seus antecessores por se valer do arcabouço de informações disponibilizadas. O uso de algoritmos baseados nos fenômenos naturais como a evolução genética, AG ou a têmpera simulada, TS, e/ou suas combinações tem sido bastante explorados...

No trabalho atual, utilizou-se uma técnica extremamente simples, porém poderosa, pertencente ao campo dos algoritmos de busca, muito utilizada na teoria da inteligência artificial, para resolver o mesmo problema. Trata-se do algoritmo de **subida de encosta** (*hill climbing*), o qual tem como principal característica encontrar boas soluções com pouquíssimo tempo de processamento...

Desta forma, associando-se o algoritmo de subida de encosta com técnicas adequadas e genuínas (heurísticas) para resolver os problemas envolvidos no sentido de encontrar uma boa solução para o cenário acima exposto, conseguiu-se resultados promissores que validaram o uso do algoritmo citado neste campo de atuação.

Palavras-Chave: Subida em encosta; Placa de Circuito Impresso, PCI; Tempo de captura; Tempo de inserção; Máquina de Inserção Automática; Escaninho; Magazine; Alimentador; Função de Custo; Busca Local; Otimizador; Ciclo de Montagem.

Abstract

This work dealt with the problem of optimizing the mounting sequence of electronic components on printed circuit boards, PCBs. The solution to this problem would be tantamount to finding the best sequence of assembly of all components of the plaque that would guarantee better productivity of this industrial activity. The solution might be found in the shortest possible time, since this would be useful in evaluating the various sequences simulated. This is the basic desire of the automatic assembly components industry, which use high-speed machinery and specialized algorithms to achieve the maximum productivity.

Many studies have already been done with this goal, and gradually achieved favorable results in comparison with its predecessors because it has used the framework of information available in these earlier studies. The use of algorithms based on natural phenomena such as genetic evolution, the GA or simulated tempering, SA, and / or their combinations, has been explored.

In this research we used an extremely simple and powerful technique used of the search algorithm area widely applicable in the theory of artificial intelligence. This is the algorithm of **hill climbing**, which has as its main feature find good solutions with very little processing time...

Thus, associating the algorithm of the hill climbing with appropriate heuristics to solve the problems involved in finding a good solution for the above scenario, it was possible to achieve promising results, validating the use of the algorithm in this field of action.

Keywords: Hill climbing**;** Printed Circuit Boards, PCBs; Surface Mount Device, SMD; Automatic Tool Changer, ATC; Surface Mount Technology, SMT; Nozzle; Feeder; Multi Head Placement Machine; Pick&Place; Trolley; Slot; Local Search;

Sumário

Índice de figuras:

Índice de Tabelas

Siglas Usadas:

AG - Algortimo Genético

AM - Algortimo Memético

ANC - **(***Automatic Nozzle Changer*)

ATC - **(***Automatic Tool Changer*) Dispositivo de Troca de Ventosas

A-TEAMS - Tempo de processamento assíncronos (*Times Assíncronos*)

AVK - Modelo de máquina de inserção da marca Panasonic

BT - Busca Tabu – algoritmo de busca

CPU - Unidade de Processamento Central (*Central Process Unit*)

HLC - Algoritmo de otimização das máquinas JUKI (*Host Line Computer*)

JUKI - Nome de fabricante de maquinas de inserção automática

MATLAB - Ferramenta matemática comercial usada em trabalhos de pesquisa científica

MSVC++ - Ferramenta gráfica de programação em linguagem c++ da *Microsoft*

NP - Não Polinomial

PCI - Placa de Circuito Impresso

RAM - Memória de Acesso Randômico (*Randomic Access Memory*)

SA - Algoritmo de busca Tempera Simulada (*Simulated Annealing*)

SMD - Dispositivo de Montagem em Superfície (*Surface Mount Device*)

SMT - Tecnologia de Montagem em Superfície (*Surface Mount Tecnology)*

TI - Tecnologia da Informação

TM - Tempo de montagem

TS - Tempera Simulada – Algoritmo de busca local

TV - Troca de Ventosas – Tempo para troca de ventosas

1.Introdução

O gargalo da indústria de produtos eletrônicos concentra-se cada vez mais no tempo necessário para a montagem de componentes sobre as placas de circuito impresso, PCIs. Diversos algoritmos, baseados em abordagens teóricas distintas, foram desenvolvidos com o objetivo de obter as melhores respostas de otimização da seqüência de montagem de componentes sobre PCIs. Esse processo de otimização visa à diminuição do tempo de montagem, o que corresponde, em geral, a diminuir ao máximo à distância percorrida nos ciclos de montagem. Denomina-se ciclo de montagem, as tarefas envolvidas desde a captura do primeiro componente até a montagem do último componente. Nesse trabalho pretende-se contribuir para a plêiade de algoritmos existentes para a otimização da seqüência de montagens de componentes em PCI. Será proposto um algoritmo baseado em métodos de busca local, mais enfaticamente, no método de subida em encosta (*hill climbing*), na tentativa de chegar a uma resposta com tempo de processamento extremamente rápido (tempo para se obter uma solução) se comparado aos métodos tradicionais que se têm empregado. Tal expectativa eleva este trabalho ao nível de competição com os algoritmos especializados que são desenvolvidos pelos fabricantes das máquinas responsáveis pelo uso das mesmas, a saber, as máquinas de inserção automática de componentes. Os próximos parágrafos, paulatinamente, esclarecerão melhor o assunto em epígrafe, contextualizando o problema e a estratégia adotada.

As placas de circuito impresso são elaboradas com material isolante, geralmente plástico (epóxi, por exemplo), sobre as quais são projetados os circuitos elétricos dos dispositivos fabricados pela indústria. Tais projetos envolvem o posicionamento de componentes como: capacitores, resistores, diodos, circuitos integrados, etc. Estes por sua vez, devem ser montados sobre as posições previamente definidas e interligados por meio de processos de soldagem, a fim de que possam atender as funções para as quais foram projetados.

A diversidade de componentes eletrônicos existentes, no que tange ao tamanho, encapsulamento, características elétricas, peso, entre outros, torna a tarefa de montagem uma tarefa assaz complexa. Em face ao atual estágio tecnológico da área, existem

componentes que só podem ser montados por processos automatizados. Há outros, porém que, devido a questões relacionadas a custos de produção, devem ser montados de forma manual. Mas há casos em que, dependendo do produto, necessita-se de 100% de automação.

Para realizar a etapa automática de montagem de componentes foram desenvolvidas máquinas especiais denominadas de máquinas de inserção automática de componentes eletrônicos. Tais equipamentos foram elaborados utilizando-se tecnologias específicas para atender a montagem de grupos de componentes. Contudo, em todas elas observa-se a presença de cinco etapas básicas:

1- Posicionamento para a retirada de um componente do alimentador;
2- A retirada do componente do alimentador;
3- Transporte do alimentador até a PCI;
4- Posicionamento para a inserção do componente sobre a PCI;
5- Inserção do componente.

Dependendo da tecnologia empregada, algumas etapas podem exigir algum processamento intermediário. Existem máquinas que possuem um alimentador único. Neste caso, os componentes a serem montados devem obedecer a uma seqüência pré-determinada de disponibilização pelo alimentador. Em outras máquinas, os alimentadores (sobre os quais são conectados os componentes) são encaixados em locais especiais, adjacentemente dispostos (formando o magazine ou *trolley*). Neste caso, os componentes de um determinado tipo são enfitados, formando rolos que são encaixados nos alimentadores. Portanto, a cada alimentador corresponde um tipo de componente. Conseqüentemente, o número de componentes distintos que podem ser inseridos por uma determinada máquina é limitado ao número de locais especiais (conhecidos como *slots* ou escaninhos) que formam o magazine. Os magazines possuem formatos lineares ou circulares e podem ser fixos ou móveis. No caso de serem fixos, quem deve se movimentar em direção aos alimentadores são os braços com os cabeçotes de captura de componentes. Já no caso de serem móveis, estes se deslocam em direção ao local onde se encontram os cabeçotes. Tais diversidades de deslocamentos envolvem características físicas que influenciam no tempo de montagem geral. Outro aspecto digno de nota diz respeito às mesas de inserção sobre as quais são

colocadas as PCIs. Existem mesas fixas em que os braços de deslocamento nas direções X e Y (direções perpendiculares sobre o plano horizontal da placa) levam os componentes até as posições de inserção sobre as mesmas. Mas há também o caso contrário, onde os cabeçotes ficam em posições fixas e que se movimenta (em ambas as direções) são as mesas com as placas a serem montadas.

Em função de obedecer ou não os passos anteriormente listados, os ciclos de montagem das máquinas insersoras podem ser classificados em seqüencial ou concorrente, de acordo com as definições a seguir:

Máquinas com ciclo seqüencial – Obedecem claramente as etapas básicas anteriormente mencionadas;

Máquinas com ciclo concorrente – Nas quais as etapas de captura e inserção, podem concorrer entre si existindo dispositivos especiais de transporte entre estas etapas ou múltiplas cabeças de inserção. O próprio deslocamento dos braços X e Y podem ser concorrentes ou seqüenciais;

Notoriamente, as máquinas com recursos concorrentes são capazes de montar componentes sobre uma mesma PCI em uma determinada seqüência de forma mais rápida. O custo benefício, entretanto não pode ser desprezado, uma vez que se trata de máquinas com tecnologias mais avançadas e, em um dado momento, de custo mais elevado.

Neste cenário, a categoria da máquina afeta claramente o tempo de ciclo de montagem. No caso seqüencial, o tempo será a soma de cada uma das etapas. No caso de uma máquina concorrente, o cálculo é mais complexo, pois se deve verificar os tempos de concorrências e suas inter-relações.

Finalmente, deve-se abordar a classificação das máquinas em dois grupos separados quanto à evolução tecnológica dos componentes e das máquinas propriamente ditas. Destaca-se em um primeiro grupo a existência de máquinas de montagem de componentes convencionais, nas quais os terminais devem transpassar os orifícios existentes sobre as placas para inserção dos mesmos. Em um segundo grupo registra-se a existência de máquinas de montagem de componentes de superfície, denominados

pela sigla inglesa SMD (*Surface Mount Device*). As primeiras são denominadas *through-hole* e são subdivididas nas categorias de inserção de componentes do tipo axiais e radiais. As segundas são as de tecnologia SMT (*Surface Mount Technology*), não havendo classificação de máquinas por tipo de componente quanto ao eixo longitudinal de inserção. Porém, existem máquinas especializadas em inserção de um determinado tipo de componente, como circuitos integrados, por exemplo.

Devido ao fenômeno da miniaturização de placas e produtos, em função do avanço tecnológico e redução de custos entre outros fatores, a tendência atual é optar-se pelas máquinas SMT. Desta forma, discorre-se um pouco mais sobre este tipo de tecnologia.

Em média, as máquinas usadas na indústria são capazes de inserir 20.000 componentes por hora o que corresponde a uma produção de 60 placas/hora com 300 componentes em cada uma. Em função da tecnologia envolvida, trata-se de máquinas mais velozes do que as *through-hole* e classificadas como máquinas do tipo captura e inserção (*pick&place)*, *chip shooters*, etc.

As máquinas SMT para montagem de componentes SMD possuem diversos fabricantes e formatos, de acordo com as necessidades específicas dos componentes e processos produtivos industriais.

Em função da grande disseminação, do crescente avanço tecnológico, dos baixos custos relativos e das altas velocidades de inserção, além da possibilidade de acesso (às máquinas) para experimentos práticos. Tendo-se em vista também, a possibilidade de realizar comparações com resultados obtidos através de outros trabalhos anteriormente realizados, optou-se pela modelagem de máquinas tipo *captura e inserção* de múltiplas cabeças, conforme ilustrado na figura 1.1.

O funcionamento básico de uma máquina de captura e inserção com múltiplas cabeças inicia-se com um ciclo de captura e inserção, repetindo-se esse ciclo até a montagem total da placa. No início os componentes são capturados pelas cabeças nos alimentadores (*feeders*), sendo que é desejável capturar o máximo número de componentes no mesmo ciclo. Isso só pode ser conseguido se os escaninhos do magazine forem ocupados de forma adjacente por alimentadores específicos, tornando possível a alimentação simultânea das cabeças de inserção. Após a captura, os braços se

movem nas direções X e Y até a posição de montagem (inserção) dos componentes capturados sobre a PCI. Portanto, existe uma seqüência de montagem que depende de qual cabeça realizará a inserção em primeiro lugar, em segundo lugar, etc. A inserção é realizada através de um deslocamento na direção vertical e sentido para baixo. Terminado o processo de inserção, os braços devem retornar ao magazine para capturar novos componentes. Este processo é repetido (ciclo após ciclo) até a montagem total dos componentes. Entre um ciclo e outro pode ser necessário realizar a troca de ventosas (*nozzles*) de uma ou mais cabeças. Estas ventosas são encaixadas nas extremidades das cabeças e são responsáveis pela captura dos componentes dos alimentadores. Tais dispositivos variam de tipo em função do tamanho, de superfície de contato e peso dos componentes. Existem máquinas que possuem uma ventosa universal para todas as cabeças, porém o caso mais comum é existirem vários tipos de ventosas. É desejável que haja o menor número de troca de ventosas possível durante o processo de montagem, pois se verifica que o tempo de troca de uma ventosa é consideravelmente alto quando comparado ao tempo de inserção propriamente dito. Na máquina JUKI modelo KE-2030, as especificações reportam que o tempo de troca chega a ser cinco vezes maior que o tempo de inserção sem troca de ventosa no mesmo ciclo. Existe um dispositivo da máquina denominado por ANC ou ATC onde são realizadas as trocas de ventosas.

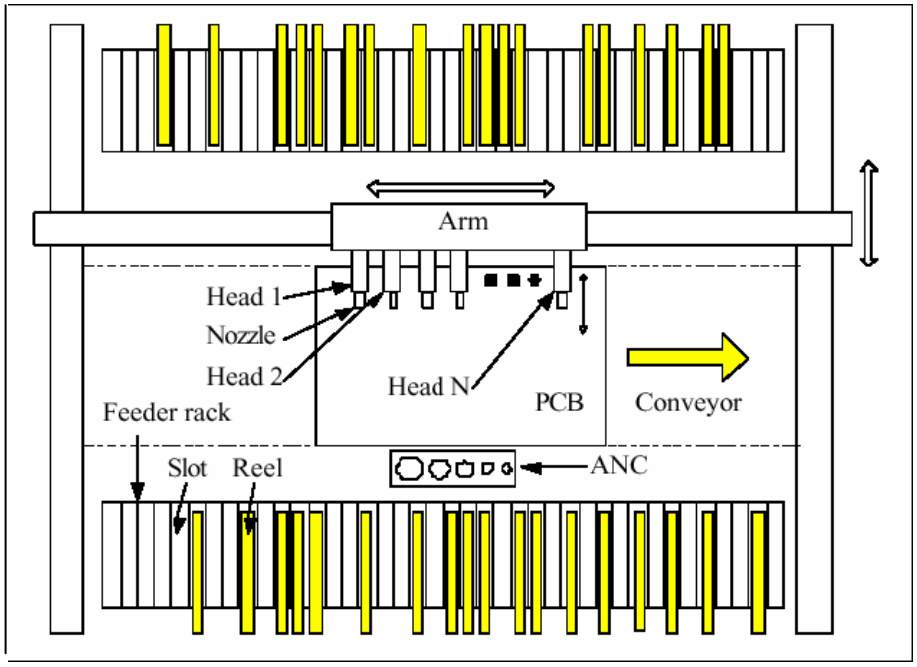

FIGURA 1.1 – Visão superior de uma máquina SMT Multi cabeças

1.1 Justificativa

A indústria sempre procurou otimizar a utilização de recursos de produção em busca da eficiência máxima e o setor de inserção automática de componentes não foge a este processo de otimização, que tem por objetivo a produção de um maior número de placas por hora. Quanto maior a produção melhor será a relação entre custos e benéficos. Desta forma, no contexto estrito de produção industrial, observa-se que a produtividade está diretamente relacionada à minimização do tempo ou distância necessária para a montagem de componentes de uma determinada PCI de um produto. O gargalo da indústria de montagem de PCIs tem se concentrado nesta etapa do processo. A diferença de um processo de montagem com tempo menor para um processo de montagem com tempo maior deve-se menos às diferenças entre máquinas de inserção e mais às seqüências de montagens oriundas dos algoritmos empregados pelas máquinas.. Assim, um trabalho que contribua para a otimização do tempo ou distância percorrida deste a captura do primeiro componente até o último componente a ser montado, vai de encontro às necessidades fundamentais dos processos produtivos do setor de inserção automática de componentes. Tal otimização é conseguida escolhendo-se a seqüência apropriada de inserção de componentes, o que corresponde a resolver uma instância do **problema do caixeiro viajante ("traveling salesman problem" ou TSP) [Lawler et al. 1985];** Ela também fará uma colocação mais apropriada de componentes nos alimentadores encaixados sobre os escaninhos dos magazines, que é visto como um **problema de alocação quadrática ("quadratic assignment problem" ou QAP) [Lawler, 1963]**. Ambos os problemas são caracterizados pela explosão combinatória e classificados como problemas NP-difíceis [Garey & Johnson, 1979]. O termo NP refere-se a problemas que não são resolvidos em tempos polinomiais. Algoritmos rigorosos para resolução deste tipo de problema são extremamente lentos e consomem muitos recursos computacionais. Já os heurísticos são sensíveis às instâncias do problema, ou seja, uns dão bons resultados e outros dão maus resultados.

Aos dois problemas acima mencionados que se relacionam com o problema de inserção automática de componentes, deve-se somar um terceiro problema, qual seja o problema **da troca das ventosas (das máquinas SMT)**. Trata-se também de um

problema caracterizado pela grande quantidade de combinações possíveis, o qual, em conjunto com os problemas de otimização da seqüência de inserção e otimização da colocação de componentes nos alimentadores, elevam ainda mais o grau de dificuldades para obtenção de uma solução ótima.

Elaborar um algoritmo original que propicie melhoria de produtividade na montagem de placas de inserção automática e com tempo de processamento pequeno (o que pode ser de grande auxílio em simulações de processos produtivos) é motivação mais do que suficiente para um trabalho de dissertação. A despeito de outros fatores existentes que também se poderiam considerar para otimização do processo, os esforços nesse trabalho se concentrarão nos três níveis (inter-relacionados) de problemas acima expostos.

De forma inovadora quanto às heurísticas, pretende-se utilizar um algoritmo de busca local, mais precisamente o de subida em encosta com reinício aleatório, para a resolução dos três problemas citados.

Nos métodos de busca local é muito útil conhecer a topologia do espaço de estados (ver figura 1.2). A topologia fornece a posição (estado) e a elevação (através do valor da *função objetivo* ou *da função de custo heurística*). Em problemas que utilizam uma *função objetivo*, o algoritmo otimiza a solução buscando o máximo local da mesma da função. Em problemas que utilizam uma *função de custo heurística*, o algoritmo otimiza as soluções buscando o mínimo local da função. Um algoritmo **ótimo** sempre encontrará o máximo ou mínimo **global** em vez do valor local.

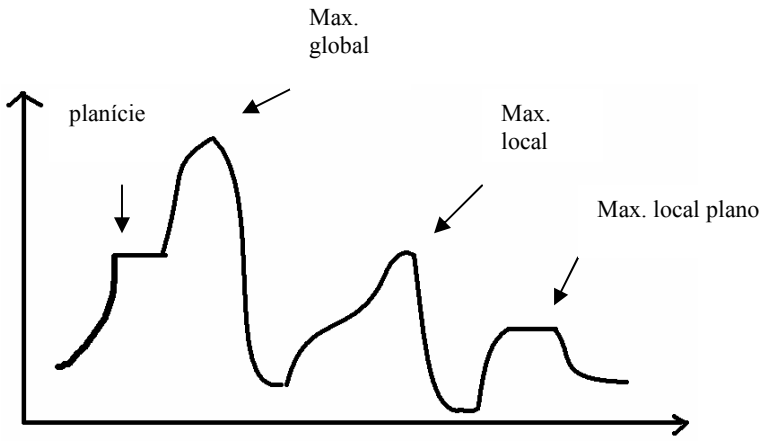

FIGURA 1.2 – Uma topologia de espaço de estados unidimensional

No caso do problema de *otimização de seqüências de montagens de componentes* o objetivo é a minimização de uma função de custo que minimiza o tempo ou a distância envolvido nas três etapas anteriormente descritas de montagem de componentes SMT.

Ilustra-se a seguir a utilização de um método de busca de força bruta e o método de busca de subida de encosta com reinício aleatório na busca de uma solução para o problema das *oito rainhas* (Russel, IA, pg. 67).

Esse problema tem como objetivo posicionar *oito rainhas* em um tabuleiro de xadrez de tal forma que nenhuma rainha ataque outra qualquer, sendo que uma rainha ataca qualquer outra situada na mesma linha, coluna ou diagonal. A figura 1.3 ilustra uma tentativa de solução que falhou. A rainha na coluna mais a direita é atacada pela rainha no canto superior esquerdo. Uma das formulações possíveis para resolução desse tipo de problema, que não utiliza busca local, é a formulação incremental. Essa formulação pode ser definida como: Escolhe-se uma posição no tabuleiro para colocar-se a primeira rainha. Para essa primeira escolha existem 64 possibilidades. Escolhe-se uma posição para colocar-se a segunda rainha. Para cada uma das 64 possibilidades anteriores existem 63 possibilidades de se colocar a segunda rainha. E assim sucessivamente.

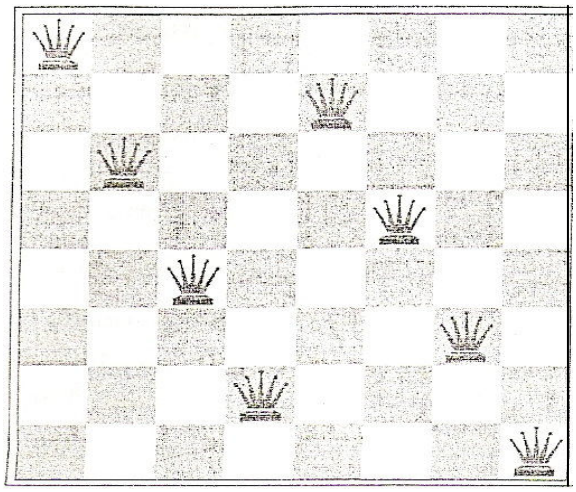

FIGURA 1.3 – (Russel, IA, pg. 68) Uma quase solução para o problema de 8 rainhas.

Nesta formulação teríamos 64 x 63 x... x 57 ≈ 3 x 10^{14} estados possíveis, o que representa um espaço de busca de dimensões consideráveis, principalmente quando o número de rainhas aumenta.

Uma outra formulação envolvendo o método de subida de encosta com reinício aleatório poderia ser definida através do seguinte algoritmo:

1) Coloca-se uma rainha em cada coluna do tabuleiro de forma aleatória;
2) Escolhe-se uma coluna. Varia-se a posição da rainha nessa coluna até atingir o menor número de ataques entre as rainhas no tabuleiro;
3) Repete-se o procedimento 2 um determinado número de vezes pré-fixado. Caso a solução não seja encontrada voltar ao passo 1 até que uma solução seja encontrada.

O algoritmo de subida de encosta com reinício aleatório tem duas características fundamentais:

1) Busca uma solução na vizinhança de um estado. No caso das oito rainhas, na vizinhança do estado aleatório inicial;
2) No caso de não encontrar uma solução, reinicia a busca na vizinhança de outro estado aleatório;
3) Utiliza apenas o estado corrente, não se preocupando com o caminho utilizado para alcançar a solução.

Conforme afirmado no item 3, o algoritmo descrito opera usando apenas um único estado (não guardam caminhos anteriores para comparação) e em geral desloca-se apenas para os *estados vizinhos* do mesmo. Esta abordagem confere ao algoritmo de subida de encosta com reinício aleatório duas vantagens interessantes:

- Usa pouquíssima memória;
- Geralmente pode encontrar soluções razoáveis em um tempo reduzido em gigantescos espaços de estados;

Além das vantagens anteriormente citadas, que justificam a escolha do método local de subida de encosta com reinício aleatório, resultados promissores da utilização do método em outras aplicações, com significativa redução de tempo na busca de uma solução, justificaram a escolha do mesmo. Como por exemplo: O uso de métodos locais com heurística de conflitos mínimos foi empregada na programação de observações do telescópio Hubble, reduzindo o tempo necessário para programar uma semana de observações de três semanas para algo em torno de 10 minutos. [Russel, 2003].

Esse trabalho estará inteiramente justificado se, ao final do mesmo, as seguintes perguntas forem respondidas a contento:

1) Será que o método subida de encosta com reinício aleatório realmente serve para ser usado na modelagem de uma máquina de inserção automática de multi-cabeças?
2) Será que o método apresentará a característica de encontrar soluções em tempos extremamente rápidos como informam trabalhos anteriores desenvolvidos com o mesmo para outras aplicações?

A hipótese levantada é que o método de subida de encosta com reinício aleatório é perfeitamente aplicável ao caso da otimização de montagem de componentes eletrônicos de PCI mantendo o seu caráter de extrema rapidez em função do que já foi exposto. O impulso inicial partiu da adaptabilidade do método ao caso anunciado. Porém, se além da otimização real com relação ao tempo de montagem, também houver otimização real em termos de tempo de processamento das análises combinatórias, então ter-se-á satisfeito a ânsia da indústria por um algoritmo eficiente e com baixos custos.

1.2 Objetivo

1.2.1 Objetivos Gerais

Apresentar e avaliar uma metodologia para otimização do tempo de inserção de componentes para máquinas SMD utilizando-se o método de subida de encosta com reinício aleatório.

1.2.2 Objetivos Específicos

- Comparar o desempenho de arquiteturas para solução do problema de inserção de componentes utilizando o método abordado;

- Comparar o desempenho da arquitetura mais promissora com o desempenho de outras técnicas de otimização já utilizadas para a solução da mesma placa padrão;

- Buscar respostas competitivas com relação a resultados obtidos com o algoritmo comercial do fabricante (HLC) para resolver um mesmo caso de estudo.

1.3 Organização do Trabalho

A primeira parte deste documento introduz o problema e discorre, resumidamente, sobre alguns trabalhos pesquisados, com a intenção de criar uma motivação para o desenvolvimento da proposta sugerida, o que corresponde aos dois primeiros capítulos. Na segunda etapa, ou terceiro capítulo, a teoria do método utilizado será minuciosamente desenvolvida, exemplificando as modelagens e suas aplicações. Após isso, no quarto capítulo, serão apresentados os resultados: comparações entre os modelos implementados; comparação do modelo mais promissor com os resultados de otimizadores de fabricantes da maquina utilizada e comparações com outros métodos anteriormente já utilizados e descritos na literatura. As conclusões da pesquisa, com tudo o que pôde e o que não pôde ser validado, serão registradas na etapa final, bem como as perspectivas para trabalhos futuros, tanto de complementação quanto de inovações.

2.Revisão Bibliográfica

Dentre os diversos trabalhos realizados para resolver o problema de otimização de tempo de montagem de placas por máquinas insersoras, destacam-se aqui, os métodos utilizando algoritmos genéticos, times-assíncronos e alguns outros.

2.1 Algoritmos Genéticos

Na literatura encontram-se diversos artigos que tratam da aplicação de algoritmos genéticos. Para esse trabalho de dissertação revisamos três aplicações distintas as quais se passam a discorrer.

Numa primeira aplicação os autores [Lee, 2000], [Burke, 1999] e [Burke, 2000], empregaram cromossomos que representam as seqüências de inserção de componentes, a distribuição nos magazines e as seqüências de montagem pelos cabeçotes em determinado ciclo. Nesse trabalho procurou-se resolver simultaneamente o problema de otimização desses três níveis. Para isso, definiram-se funções capazes de avaliar os tempos de inserção obtidos pelas gerações oriundas dos cruzamentos e mutações entre aqueles cromossomos. O objetivo sempre é minimizar o tempo total envolvido na montagem.

Em outro trabalho utilizando algoritmo genético, [Craveiro, 2005], houve a separação do problema de otimização de troca de ventosas (*nozzles*) pelas cabeças de inserção. Nesse trabalho, elaborou-se o modelo para um algoritmo genético específico, constituído de duas partes. Na primeira parte foi realizada uma pré-otimização que consistiu em um algoritmo genético exclusivamente construído para resolver o problema do melhor arranjo de ventosas possível. O resultado desta pré-otimização foi então apresentado ao algoritmo genético principal, para que este resolvesse o restante do problema, otimizando a seqüência de montagem e de distribuição dos rolos de componentes nos escaninhos do magazine, visando encontrar o menor tempo de inserção possível. Os resultados disponibilizados pelo trabalho reportam que o tempo de processamento para encontrar uma solução na montagem de uma quantidade de

apenas 21 componentes, usando uma população de 150 indivíduos e 50 gerações, foi de 96 segundos (após uso de processos de vetorização disponibilizados pelo programa MATLAB) para uma máquina SMT de uma só cabeça de inserção. Porém, para uma máquina com quatro cabeças e aumentando-se a quantidade de componentes para 44, após 26 minutos de processamento, encontrou-se uma solução satisfatória. Estes dados sinalizam para o fato de que o algoritmo proposto pelo autor possui tempo de processamento no mínimo diretamente proporcional ao número de combinações possíveis entre os componentes.

2.2 Algoritmos Meméticos

Uma outra abordagem mais recente faz uso de algoritmos meméticos que são parecidos com algoritmos genéticos - AM, porém suas particularidades permitem classificá-los como constituintes de uma meta-heurística distinta.

Essas particularidades reportam-se basicamente a inserção de técnicas de busca local entre as etapas de um algoritmo genético visando refinar os candidatos gerados pelo mesmo. Por exemplo, após a geração da população inicial, usar-se-ia um método de busca local para selecionar os mais aptos para prosseguir no processo, qual seja, o de aplicar os operadores genéticos (cruzamento, mutação), criando-se uma nova geração que seria novamente aplicada a um novo processo de busca representando um novo refino da população. Neste sentido, os métodos de busca conduziriam a melhores aptidões para encontrar (igualmente) melhores soluções para um determinado problema.

O trabalho desenvolvido por [Carvalho, Elidelson, 2007] utilizou um AM para solucionar o mesmo problema apresentado no item 2.1. Ou seja, utilizou à mesma lista de 44 componentes apresentada de forma randômica a entrada do AM para que este encontrasse qual a melhor seqüência de montagem dos mesmos. Para isso, elegeu como função de adaptação (função objetivo ou função de custo) o tempo necessário pata tal. Portanto, quanto menor o tempo, melhor a solução.

Os métodos de busca utilizados foram: Busca tabu - BT e têmpera simulada - TS. Sendo que seus resultados reportaram que o uso do método da têmpera simulada obteve melhor resposta, sendo capaz de montar os 44 componentes em 29,5 segundos usando-se o mesmo tipo de máquina (JUKI 2030). Mas não deixou claro quanto tempo de processamento foi envolvido na solução. Contudo, tratava-se de um tempo relativamente elevado. A pesquisa também fez uso de ferramentas de AG do MATLAB para realização de simulações.

2.3 Times Assíncronos

Uma abordagem diferente e mais antiga faz uso do conceito de "*times assíncronos*" , *A-Teams*, para otimização do processo de inserção automática de componentes eletrônicos. Entretanto, esta abordagem procura otimizar apenas os dois níveis mais primitivos dos processos de inserção automática, quais sejam, a seqüência de montagem de componentes e a distribuição de alimentadores pelos *escaninhos* do magazine.

Esta técnica desenvolvida por [Scarpini, 1999], foi utilizada sobre máquinas do tipo *through hole*, com capacidade de memória para até 2000 posições e dois magazines com capacidade para 60 *escaninhos* cada. Embora formulação matemática não seja exibida no texto do artigo, conclui-se tratar-se de um problema de otimização discreta abordado por algoritmos heurísticos a época. A questão era organizar os algoritmos (para a seqüência de inserção e para a distribuição nos *magazines*) de tal forma que cooperassem entre si obtendo soluções que não seriam encontradas isoladamente [Souza & Talukdar, 1993].

O conceito de *A-Teams* é baseado na utilização simultânea e assíncrona de diversos agentes, formados por algoritmos que cooperam entre si, ou seja, os algoritmos são os próprios agentes. Sendo assim, são adaptados agentes para resolução dos TSPs (problemas do caixeiro viajante usados para melhorar as seqüências de montagens de componentes) e outros agentes para resolução de QADs (problemas de alocação quadrática usados para otimizar a colocação de alimentadores nos escaninhos dos magazines). Estes agentes são autônomos e se comunicam (os TSPs e QADs) através

de uma área de memória (arquivos) com as melhores soluções para as instâncias a resolver.

No trabalho desenvolvido pelo grupo do Laboratório de Técnicas Inteligentes, LTI, da Escola Politécnica da Universidade de São Paulo, foi desenvolvido um programa denominado por OPTIMA, no qual, usaram-se dos agentes para resolver o problema de TSP e outros dois para resolver o problema de QAD. Nos primeiros, foram usados implementações do algoritmo *Lin-Kernighan* [Reinelt, 1994] e do algoritmo *Greedy* [Rardin,1998]. No caso do segundo, foram usadas implementações do CRAFT [Armour & Buffa, 1963] e o HGW [West, 1983], ambos se tratando de algoritmos heurísticos por melhoria.

Os resultados mostraram que a abordagem de combinar QAD com TSP usando *A-Teams*, consegue obter uma melhora adicional nos tempos de inserção de 4,7% em média, em relação a uma abordagem que usa somente TSP para máquinas do tipo AVK (*through-hole* de fabricação da Panasonic). Ressalta-se, no entanto, que ele produz uma melhoria de 11,1% em média, se comparado a entradas iniciais e aleatórias.

2.4 Planejamento Hierárquico

Existe um trabalho desenvolvido por [Zeng You-jiao, MA Deng-zhe, JIN Ye, YAN Jun-qi, 2004], denominado por "*Hierarchical planning for a surface mounting machine placement*", que aborda o problema de otimização, com objetivos que estão em sintonia com o que se pretende realizar nesse trabalho de dissertação. Trata-se de um procedimento subdividido em quatro etapas, a saber:

1) Uma rotina capaz de converter as coordenadas de PCI fornecidas em arquivo gerado por uma ferramenta de CAD para o formato de dados entendidos pela máquina SMT;

2) Uma rotina capaz de selecionar os cabeçotes de forma a minimizar o tempo de troca de ventosas (*Nozzles)*;

3) Um algoritmo capaz de minimizar o tempo total de captura de componentes através da maximização de capturas simultâneas dos mesmos nos alimentadores (*feeders)* – aperfeiçoando o arranjo de *alimentadores*;

4) Uma heurística capaz de reduzir o tempo de captura e montagem (*pick-and-place time)* – aperfeiçoando a seqüência de montagem.

Estas quatro etapas também são abrangidas pela proposta da dissertação ora apresentada, porém, usando-se outras estratégias. Em trabalhos anteriores, para abordar os ítens 3 e 4, fez-se o uso de modelos de caixeiro viajante ou "*travelling salesman problem* – TSP. Em outros, o item 3 foi tratado como um problema de combinação quadrático ou "*Quadratic assignment problem*" e o item 4 como um "*3-D asymmetric TSP*" [Nevalainen – 1989].

Zeng desenvolveu o trabalho para considerar o problema de otimização em máquinas de múltiplas cabeças através do uso de procedimentos planejados, ou seja, elaborou, primeiramente, o procedimento de conversão de coordenadas de CAD para coordenadas de máquina, reduzindo desta maneira o tempo de configuração das máquinas (*setup time*); Após isso, criou um procedimento separado, para escolher os *cabeçotes* de forma que garantisse a minimização da necessidade de suas trocas durante a montagem de componentes de uma PCI. Para tal, lançou mão de um algoritmo com algumas restrições, sendo a principal delas a seguinte: Deve haver um e somente um tipo de *cabeçote* escolhido por cada tipo de componente. Na seqüência, o seu trabalho gerou um procedimento de ordenação de *alimentadores* sobre os *escaninhos* do magazine. O objetivo deste arranjo era o de obter o menor tempo possível durante as viagens para capturar todos os componentes dos *alimentadores*. Seu procedimento baseou-se em uma função objetivo (uma expressão matemática) com várias restrições, sendo a principal a de que só deve haver um *alimentador* por tipo de componente, ou seja, não pode existir mais de um *alimentador* com o mesmo tipo de componente alocado em um outro *escaninho* (dispositivo onde é encaixado um *alimentador*). Adverte-se, contudo, que a posição dos *alimentadores* e a posterior seqüência de montagem afetam a eficiência geral de montagem da máquina. Com este pensamento, elaborou-se um procedimento onde se maximizou a captura de componentes simultâneos através das múltiplas cabeças da máquina. Com isso, a seqüência de

montagem, ficaria notoriamente afetada por essas adjacências. Finalmente, através de nova expressão matemática, determinou a seqüência de montagem final. A função objetivo inclui também, os tempos de deslocamento entre ciclos. Trata-se de uma função com várias restrições e que é modelada como um problema do caixeiro viajante, usando-se heurísticas para definir os vizinhos mais próximos.

Foram realizadas comparações de desempenho, tendo como parâmetro a eficiência obtida entre um algoritmo que realizasse otimizações de *alimentadores*, mas não realizasse otimizações nas seqüências de montagens utilizando o procedimento planejado conforme resumido acima. A eficiência dos algoritmos foi calculada em relação aos tempos de montagem final de uma mesma PCI. O artigo revela que se obteve uma melhoria de eficiência da ordem de até 45,94% a favor do Planejamento hierárquico.

2.5 Planejamento em tempo real para Máquinas de Inserção do tipo Múltiplas Cabeças

Abordaremos a seguir, um resumo sobre o trabalho desenvolvido pelo *Automated Scheduling, Optimisation and Planning (*ASOP*) Research Group*, da Universidade de Nottingham, Reino Unido, Escola de Ciências da Computação e TI [Ayob & Kendall, 2003].

A pesquisa se dispôs a aproveitar o avanço tecnológico representado por um dispositivo de alimentador considerado inteligente por ser capaz de identificar as posições sobre os *escaninhos,* dos tipos de componentes utilizados durante o processo de captura e montagem de componentes SMD sobre PCIs. Este dispositivo também é capaz de detectar a ausência de componentes de um determinado tipo durante o processo de montagem. Desta forma, o trabalho visou desenvolver uma heurística usando um algoritmo subida de encosta para achar uma solução inicial que aperfeiçoasse a seqüência de captura e montagem dos componentes sobre a PCI. Após encontrar a solução inicial, e tendo-se em mente que os componentes podem ser misturados por engano durante o processo de carregamento dos mesmos sobre

alimentadores-escaninhos, ou que os mesmos podem acabar durante a montagem, inviabilizando a solução inicial, a heurística deverá encontrar nova solução levando-se em conta estas alterações dinâmicas. Porém, esta nova solução deve consumir apenas o tempo ocioso da CPU da máquina, ou seja, os intervalos de tempo existentes durante os movimentos dos braços do robô ao capturar ou montar um componente. A utilização de tempo ocioso confere ao método um caráter de tempo real, pois não interferiria no processo de montagem do ciclo atual durante a busca de uma nova solução. Resultados experimentais reportaram melhorias significativas nas soluções.

Os autores destacam que a maioria dos trabalhos desenvolvidos no sentido de otimizar a seqüência de inserção possuem um caráter de tempo não real, ou seja, as soluções encontradas são inerentes a uma distribuição de componentes pré-determinada e que não se alteraria durante todo o processo de montagem. Além disso, o tempo despendido para encontrá-la em geral é alto. A pesquisa abordada, ao contrário, além de permitir uma constante atualização da solução, o faz em períodos de tempo bastante resumidos.

A atualização da solução, contudo, só é possível em função do recurso tecnológico para detecção de componentes por dispositivos *alimentadores* considerados *inteligentes ou espertos*. Máquinas do tipo MY15 e MY19 possuem este tipo de característica e as experiências realizadas envolveram este tipo de máquina. Já a rapidez da convergência, pode ser atribuída ao uso de heurísticas associadas ao método de subida de encosta.

Usa-se o recurso de interrupção de hardware como método, através de sensores, para aproveitar o tempo ocioso da CPU durante as operações de captura, montagem e movimento dos braços do robô. Entre estes eventos de interrupção é disparado o algoritmo de busca de subida de encosta. Experiências realizadas usando-se dois conjuntos de dados iniciais, ou seja, duas seqüências de captura e montagem distintas, através das quais se aplicou o método proposto, revelaram que houve uma melhoria de 58,79% e 76,69%, respectivamente, com relação às soluções iniciais (seqüências iniciais).

2.5.1 Resumo do processo

Utilizou-se uma máquina do tipo múltiplas cabeças com dispositivo para detecção de *alimentadores* inteligente, mesa de colocação de PCI fixa, robô com braços x e y para captura e montagem de componentes, *cabeçotes* para capturar os componentes. A solução inicial foi obtida através de um método de busca local e o processo de montagem usa esta solução para montar todos os componentes sobre a PCI. Entre os eventos de captura, montagem ou movimento dos braços, uma interrupção é disparada e, usando-se um algoritmo subida de encosta que aceita apenas soluções melhores, uma nova solução pode ser disponibilizada, para a montagem de uma nova PCI. Por se tratar de um algoritmo subida de encosta, o tempo de convergência é extremamente rápido (o artigo reporta tempos da ordem de segundos). Vários fatores influenciam a escolha de uma nova solução. Entre eles: O agrupamento de componentes para um mesmo ciclo; o arranjo para montagem e o arranjo para a captura de componentes; o próprio arranjo de *ventosas*, posto que as trocas de *ventosas* acarretam em acréscimos consideráveis de tempo de montagem final; além de outros fatores. A melhor equalização desses diversos fatores deve estar contemplada dentro da heurística usada.

Na heurística adotada houve trocas entre componentes de um mesmo ciclo de captura ou de montagem, bem como entre os ciclos da solução. O caráter de execução das rotinas por meio de interrupções para processar esta heurística garante ao método que ele não acrescentará qualquer fração de tempo ao tempo de montagem final da PCI. Então, se houver necessidade, a CPU da máquina onde as rotinas são processadas, pode ser interrompida paralisando o processo para que o mesmo possa ser retomado durante o próximo período de tempo ocioso da mesma. Além disso, se houver alguma alteração nas condições inicias, como a falta de algum componente ou o carregamento errado dos mesmos pelo operador, o sistema poderá, da mesma forma, fornecer uma nova solução inicial e continuar o processo de otimização contínua para as demais montagens de PCIs. Desta forma, não haveria a necessidade de parar o processo de montagem para corrigir o problema, ou despender um tempo demasiadamente alto para a obtenção de uma nova solução considerada ótima ou quase ótima como é o caso da maioria dos sistemas desenvolvidos.

2.5.2 Implementação

O processo inicia pela detecção de todos os componentes que estão disponíveis para a montagem sobre a PCI, ou seja, somente aqueles tipos de componentes presentes e carregados nos alimentadores sobre os escaninhos. Caso haja disponibilização de novos componentes durante o processo uma nova solução inicial será gerada.

Aleatoriamente serão atribuídos componentes dentro de um ciclo de montagem e ou captura. O tamanho e a quantidade de ciclos para a montagem total dos componentes sobre a PCI dependem do número de *ventosas* disponíveis por cabeça. Melhor dizendo, depende de quantas *ventosas* podem ser carregados simultaneamente sobre o dispositivo de múltiplas cabeças. Cada ciclo consiste de um conjunto de pontos de captura e de montagem de componentes, sendo que as *ventosas* usadas para capturar ou montar estes componentes são indexados de zero (para *ventosas* mais a esquerda) até G-1 (mais a direita). Onde G representa o número de ventosas no ciclo.

Para otimizar a atribuição aleatória inicial são aplicadas as seguintes heurísticas:

1) G trocas entre o $i^{ésimo}$ componente e o $j^{ésimo}$ componente (aleatoriamente selecionados) no mesmo ciclo de captura;

2) G trocas entre o $i^{ésimo}$ componente e o $j^{ésimo}$ componente (aleatoriamente selecionados) no mesmo ciclo de montagem;

3) G trocas entre o $i^{ésimo}$ *nozzle* e o $j^{ésimo}$ *nozzle* (aleatoriamente selecionados) no mesmo ciclo de captura; Se a troca envolver componentes de tipos diferentes, então se deve providenciar a troca de *nozzle* que garanta a correta montagem de um componente sobre o PCI;

4) G trocas entre o $i^{ésimo}$ *nozzle* e o $j^{ésimo}$ *nozzle* randomicamente selecionado de um mesmo ciclo de montagem; Se a troca envolver componentes de tipos diferentes, então, deve-se providenciar a troca de *nozzle* que garanta a correta captura de um componente sobre o PCI;

5) Q trocas entre pontos de montagem de componentes de dois ciclos de montagem, onde Q representa o número total de pontos de montagem de componentes que estão disponíveis sobre a PCI. Cada ponto de montagem em um ciclo será trocado com o outro ponto de outro ciclo randomicamente escolhido. Se a troca envolver dois tipos de componentes diferentes, então um outro componente de um apropriado ciclo deve ser adequadamente capturado de tal forma garantindo que os componentes capturados em ambos os ciclos que participaram da troca sejam válidos.

6) S trocas entre os ciclos de montagem, onde S representa o número de ciclos de captura-montagem (*pick and place*).

Os autores consideraram que não havendo uma estratégia pré-determinada, o tipo de troca escolhida durante o processo de busca de solução, seria randomicamente escolhido através da geração de um número inteiro entre 1 e 6. O processo pararia quando o número de iterações chegasse a 100.

2.5.3 Resultados e Conclusões do método apresentado no início deste item

Os autores do projeto fizeram uma série de considerações relativas aos parâmetros que alimentam o simulador criado. Dentre as mais destacadas poderíamos citar:

1) A simulação considerou que a máquina teria o dispositivo de alimentadores-escaninhos, bem como a mesa de colocação da PCI em posições fixas. Além disso, estes dispositivos estariam tão próximos quanto seria necessário para garantir que maior distancia percorrida pelo braço robótico durante o processo de captura e montagem de componente seria de 10 unidades de comprimento;

2) Foi considerado que o número de pontos de montagem seria de 20 em 100 possíveis sobre a PCI e que o número de componentes diferentes seria de 8 do total de 20 montados para um determinado conjunto a ser simulado;

3) Além disso, o comprimento da PCI seria de 100 unidades de distância e a largura de 50 unidades;

4) Também foi definido um valor de 10 unidades de comprimento por unidade de tempo, como sendo a velocidade de deslocamento do braço e que todos os componentes seriam capturados e montados com esta mesma velocidade, ou seja, velocidade constante;

5) Considerou-se que o número de *ventosas* por ciclo seria de 4.

6) Todas as gerações de ciclos de montagem e de captura, bem como as escolhas de *ventosas* por ciclo e ordenamento de ciclos para a montagem da PCI, seriam aleatoriamente geradas. Também se criou uma heurística para agrupamento de componentes que fazem parte de um ciclo.

7) Para os dois conjuntos de dados mencionados a seguir, foi considerado que todos os componentes usavam o mesmo tipo de *ventosa*.

8) Nas simulações foi usado um computador com CPU AMD *Athlon* XP1700+ PC com 1,47GHz de velocidade e 240 MB de RAM.

9) Maiores detalhes dos resultados das simulações podem ser encontrados no artigo original.

Com tais considerações criaram-se dois conjuntos de dados para estes. Um conjunto A, com 20 componentes sendo 8 de tipos deferentes e um outro conjunto B, com 100 componentes sendo 20 de tipos diferentes. Para cada conjunto, e através do processo randômico acima mencionado, geraram-se suas respectivas soluções iniciais.

O objetivo de definir dois conjuntos seria o de demonstrar que o método é eficaz tanto para um conjunto pequeno quanto para um conjunto grande de componentes.

Durante testes constatou-se que uma solução inicial foi obtida em 0,070 segundos para o conjunto A, e em 0,236 segundos para o conjunto B; Que houve uma melhoria em relação a solução inicial de 58,79% com tempo de processamento do algoritmo subida de encosta de 10,22 segundos em média, para o caso do conjunto A. Já para o caso do conjunto B, houve uma melhoria de 76,67% após 75,703 segundos de processamento. Os resultados também mostraram que uma boa solução inicial não é garantia de uma melhor solução final com relação a uma outra solução inicial que seja pior do que a primeira. Por exemplo, durante os testes foram realizadas 10 simulações tanto para o conjunto A quanto par o conjunto B, e que, durante o teste 2 do conjunto A partiu-se com um CT=54,23 unidades de tempo e terminou com um CT=28 unidades de tempo. Porém o teste 5 do mesmo conjunto A, iniciou com um CT=49,11 unidades de tempo e terminou com um CT=38,54 unidades de tempo. Resultados semelhantes foram encontrados para o conjunto B, embora este conjunto seja bem mais complexo que o conjunto A, devido a maior quantidade de combinações possíveis para o processo de montagem.

A conclusão para o método abordado é de que se trata de um método válido em função de haver tempo ocioso de trabalho da CPU que controla a máquina. Este tempo pode ser aproveitado para disparar o processo de heurística usando o método subida de encosta (busca local). Também se pôde abstrair que a qualidade da solução final é independente da solução inicial.

2.6 Comentários sobre os métodos citados

Com relação aos trabalhos elaborados em AG e AM, por [Craveiro, 2004 e Carvalho, 2007], respectivamente comentados acima, apesar dos bons resultados reportados, ficou um sentimento de que o tempo de processamento dos mesmos poderia ser reduzido significativamente, uma vez que tais métodos envolvem ferramentas matemáticas poderosas como o MATLAB, o qual lança mão de expressões matemáticas complexas para programar modelos genéticos e afins. Por outro lado, a abordagem sistemática das explosões combinatórias envolvidas, torna este tipo de procedimento relativamente lento.

Um outro fato digno de nota, diz respeito à afirmação dos autores de que o uso de apenas um *escaninho* por tipo de componente representaria uma desvantagem metodológica destes trabalhos em relação à possibilidade de haver mais de um *escaninho* com o mesmo tipo de componente. Os autores afirmaram que isso seria um dos motivos pelos quais os resultados obtidos em termos de tempo de montagem da mesma PCI foram aproximadamente 3% superiores aos encontrados pelo algoritmo comercial HLC, por exemplo. Contudo, isso não foi devidamente provado, pois, a princípio, o que pode à primeira vista parecer uma desvantagem, num segundo momento poderia ser considerado como uma vantagem, haja visto que o espaço de busca estaria reduzido (portanto, havendo maior probabilidade de se percorrer um percentual maior do mesmo em busca da melhor solução). Além disso, existe uma quantidade limitada de *escaninhos*, portanto, nada impediria que houvesse recombinações dos pares *escaninhos-componentes*, selecionando-se de forma natural, o melhor par, no caso de existirem dois ou mais pares com o mesmo componente deixando o outro de fora.

Quanto ao modelo *A-Teams*, de Scarpini, 1999, trata-se de uma abordagem mais antiga, mas que também faz uso de ferramentas pré-existentes, e desta forma, talvez menos flexíveis, mas bem adaptadas às máquinas *Through-hole*. O processamento simultâneo de algoritmos, provavelmente leva a resultados com menor tempo de processamento. Contudo, abrange apenas os dois níveis mais básicos da otimização,

deixando de lado a otimização de troca de *cabeçotes*, menos comum à época de sua elaboração.

Já o método abordado Zeng, parece ser mais completo e abrangente, tanto em relação à resolução dos vários níveis de otimização, quanto em relação à profundidade de exploração das explosões combinatórias inerentes. Porém, por tratar-se de uma metodologia estritamente baseada em expressões matemáticas de difícil tratamento, também conduz a uma sensação de grande consumo de tempo de processamento, provavelmente geométrico. De qualquer forma, o autor não esclareceu esta questão.

Finalmente, cabe ressaltar que a metodologia elaborada pela equipe Universidade de Nottingham, Reino Unido, Escola de Ciências da Computação e TI, o não foi usada durante o desenvolvimento do trabalho atual. No máximo, aproveitou-se parte da notação matemática conforme exibida durante o discorrer deste texto. Contudo, houve muitas convergências entre as idéias utilizadas. De outro lado, de forma alguma houve sobreposição metodológica ou contexto usado. Ademais, o método denominado por *Real-time Scheduling for Multi Headed Placement Machine*, no final, apresentou apenas resultados de simulação e fez uma série de restrições que não foram necessárias durante o desenvolvimento da dissertação corrente. Como exemplo, pode-se citar que durante a simulação, os autores consideraram que todos os componentes utilizam o mesmo tipo de *ventosa*, o que simplifica bastante a aplicação e conseqüente tempo de convergência na busca de melhores soluções. Outra questão, diz respeito às considerações que tornaram a modelagem mais simples e possível de ser usada em um simulador como foi o caso da definição de uma velocidade fictícia para a velocidade de deslocamento e a pré-determinação das coordenadas sobre a PCI. Durante o desenvolvimento desta dissertação, observou-se que adequar o simulador às condições reais envolve um esforço extra no sentido de obter dados mais precisos para o parâmetro da velocidade e, principalmente, do controle real sobre os sistemas de coordenadas presentes entre máquina, PCI e dispositivo de *alimentadores-escaninhos*. Resumindo, para maior confiabilidade do método seria necessário confirmar resultados originados por casos reais. Para finalizar, o objetivo daquele trabalho era validar o método para o aproveitamento de tempo ocioso de unidades de processamento de dados embutidas nas máquinas insersoras. Porém, as simulações foram realizadas em computadores portáteis (ou *desktop*), não se observando dados relativos a resultados práticos reais.

3. Metodologia

A metodologia a ser apresentada nesse capítulo, embora de caráter geral no que diz respeito à solução do problema de inserção de componentes em placas de circuito impresso, teve como foco o problema de inserção automática em máquinas do tipo captura e inserção com múltiplas cabeças com tecnologia de montagem em superfície de componentes (*Pick&Place MultHead, SMT/SMD)*. De forma específica será otimizado o problema da inserção de componentes da máquina da marca JUKI, modelo 2030 composta por:

a) Um bloco com quatro cabeças (*head*) de inserção;

b) Até dois magazines (*trolley*) lineares para dispor escaninhos com alimentadores de componentes (escaninhos-alimentadores) adjacentes; Sendo cada qual com no máximo 30 *escaninhos*;

c) Uma plataforma fixa para posicionamento de PCI;

d) Um dispositivo de ATC em formato linear para até vinte e uma ventosas de captura de componentes;

e) Dois braços independentes para transporte do bloco; Sendo um no eixo x e outro no eixo y;

Considerou-se que a velocidade de deslocamento dos braços em conjunto era constante e fizeram-se análises e simulações sobre o magazine anterior apenas (a figura 3.1 ilustra as posições dos magazines em relação a máquina de inserção). Tais simplificações, contudo, foram apenas para facilitar o manejo prático do sistema, não havendo prejuízo para sua aplicabilidade usando-se ambos os magazines ou considerando-se velocidades variáveis.

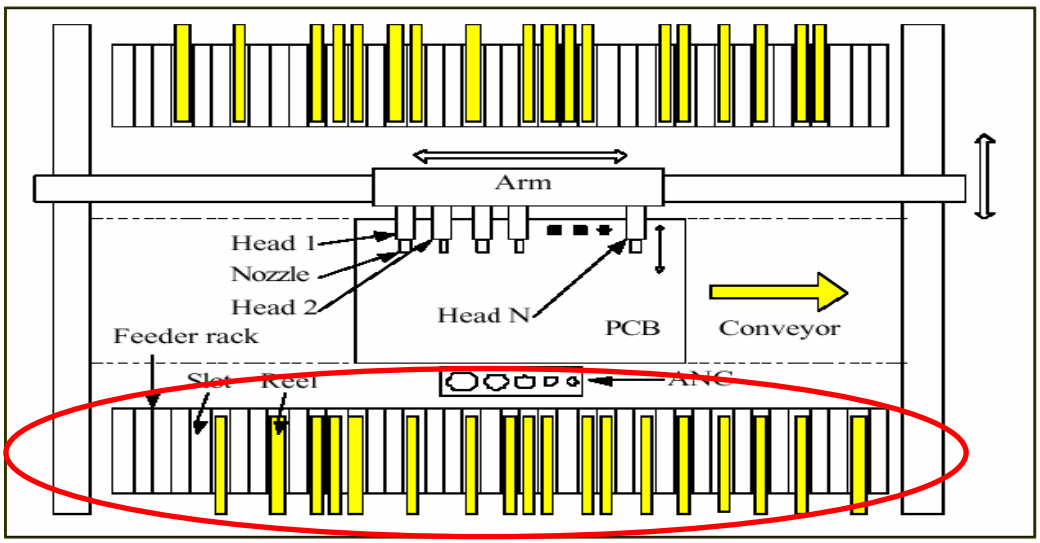

Figura 3.1 – Destaque para magazine de escaninhos frontal.

Serão avaliados dois métodos heurísticos para aplicação do algoritmo de subida de encosta generalizado com reinício aleatório. Esses métodos serão denominados a seguir de método 1 e método 2.

Os principais problemas a serem resolvidos para a aplicação do algoritmo de subida em encosta são descritos no quadro abaixo:

Problemas para aplicação do método de subida em encosta
- Definir uma representação para um estado do sistema; **- Definir um método aleatório de geração de estados iniciais;** **- Definir como serão encontrados os vizinhos de um estado;** **- Definir os critérios de parada e cálculo dos valores para a função objetivo.**

Os dois métodos propostos nesse trabalho para aplicação do algoritmo de subida em encosta ao problema de otimização da inserção de componentes em PCI trabalham com a mesma definição para o estado do sistema. Como será mostrado, o estado é

definido pela seqüência de componentes nos alimentadores, também denominada por seqüência de alimentação, e pela seqüência de montagem dos componentes, também denominada por seqüência de montagem. A diferença entre os métodos reside, como será visto, na possibilidade ou não de ter ventosas repetidas em um mesmo ciclo de montagem. No primeiro método, não se permite a repetição de ventosas. No segundo método é permitida a repetição de ventosas. Apresentam-se, a seguir, esses dois métodos.

3.1 Métodos heurísticos para aplicação do algoritmo de subida de encosta com reinício aleatório

A descrição dos métodos será realizada em três etapas: Será mostrada uma lista de etapas, um fluxograma e uma descrição detalhada das etapas.

3.1.1 Método heurístico 1

A seguir mostra-se uma lista das etapas do primeiro método proposto para resolver o problema de inserção de componentes.

1) Gerar duas seqüências de números aleatórias, dividida em duas partes, conforme mostrado na figura 3.2. *A primeira seqüência corresponde aos componentes nos alimentadores, ou seqüência de alimentação. A segunda seqüência corresponde a montagem dos componentes na PCI. Essas duas seqüências aleatórias definem o estado do problema no método 1.*

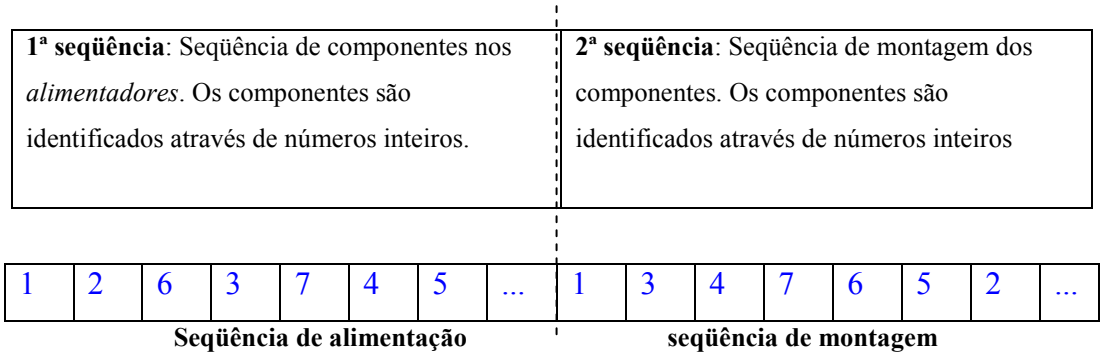

Figura 3.2: Seqüências aleatórias de componentes nos alimentadores e na montagem da PCI

2) Definir uma distribuição de ventosas para a montagem dos componentes, conforme ilustrado na figura 3.3;

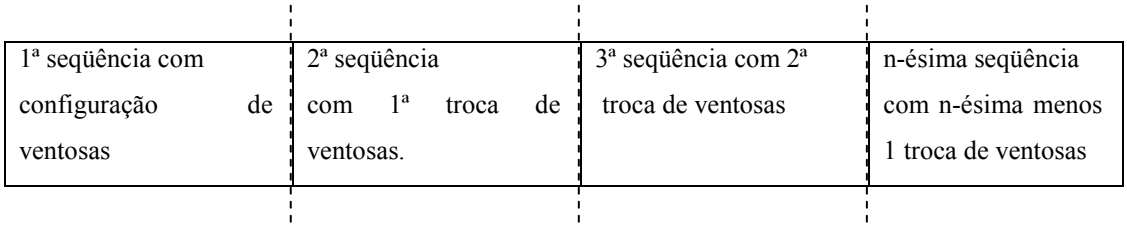

Figura 3.3: Distribuição de ventosas para montagem dos componentes

Em cada troca as 4 ventosas são trocadas. (proposta inicial)

3) Calcular a distância ou tempo baseado na função objetivo do modelo definido;

4) Encontrar um vizinho do estado presente;

5) Voltar ao passo 2 até que um critério de parada seja atingido;

6) Voltar ao passo 1 para definir um novo estado para o problema a até que um critério de parada seja atingido;
7) Mostrar gráficos com as soluções (baseadas no cálculo da função objetivo) versus a época do algoritmo;

Na figura 3.4, mostra-se um fluxograma desse primeiro método. O valor de *to* refere-se ao tempo da melhor montagem. Inicialmente o mesmo assume um valor muito grande, representado pelo símbolo ∞. Os parâmetros *n* e *m* controlam o número de estados gerados e o número de vizinhos gerados para um estado. Os parâmetros *Lim1* e *Lim2* estabelecem o número máximo de vizinhos gerados para um estado e o número máximo de estados gerados.

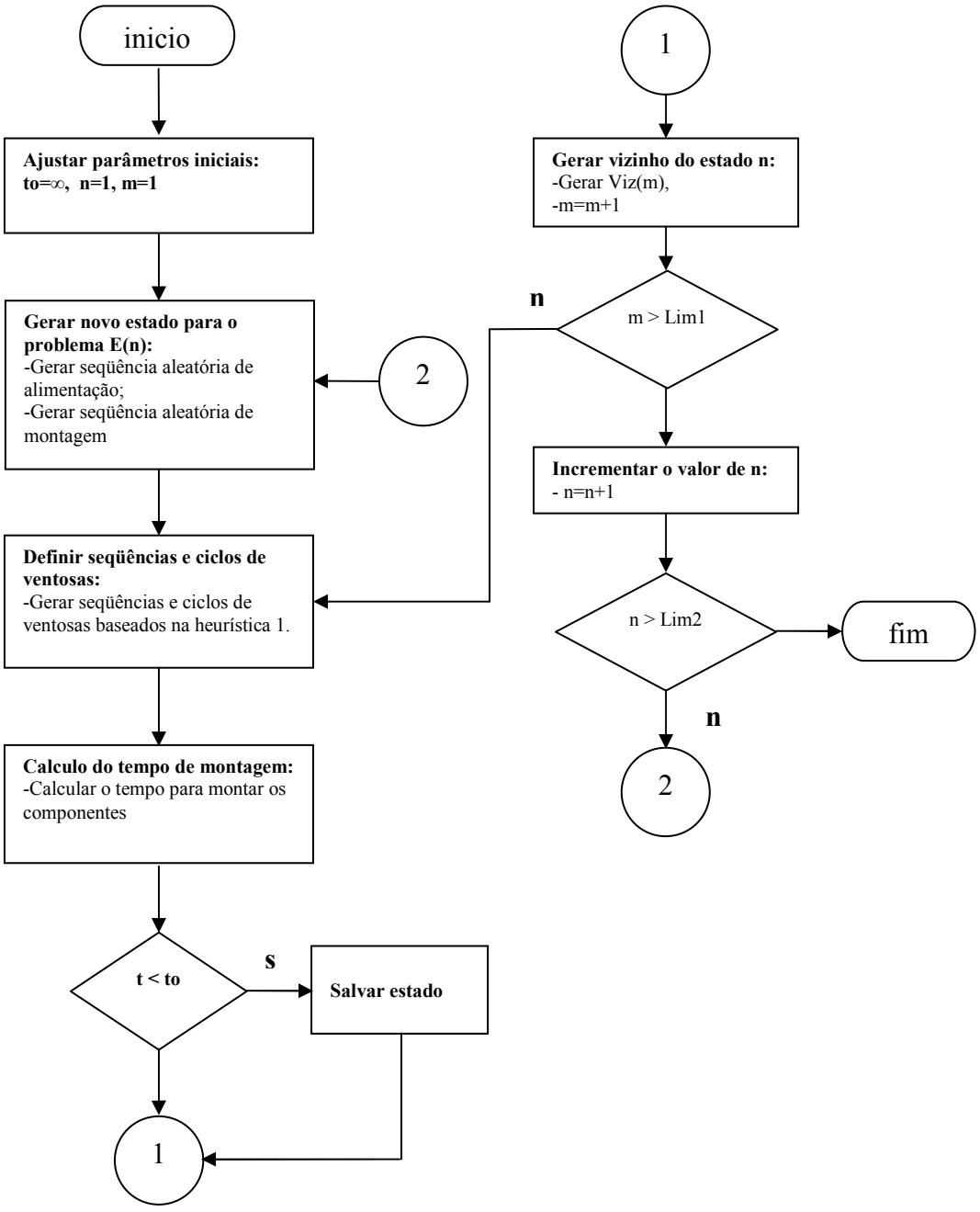

Figura 3.4: Fluxograma para a heurística 1.

A geração de um novo estado para o problema, especificado no bloco *Gerar novo estado para o problema E(n)*, é feita através da geração de duas seqüências aleatórias de números. A primeira seqüência contém a ordem de disposição dos componentes nos alimentadores. A segunda seqüência contém a ordem de montagem dos componentes.

A proposta desse método apóia-se sobre a premissa de que a escolha das ventosas é feita posteriormente à definição do estado. A seguir, ver-se-á com mais detalhes as operações realizadas no bloco *definir seqüências e ciclos de ventosas*, mostrado no fluxograma da figura 3.4.

Após a geração de um estado inicia-se o processo de definição das seqüências de ventosas utilizadas no processo de montagem dos componentes, conforme mostrado na figura 3.4. Define-se inicialmente uma *seqüência 1* de *ventosas* necessária para montar o conjunto dos *n* primeiros componentes da seqüência de montagem. O valor de *n* pode assumir um valor mínimo igual a 4 e um valor máximo igual ao número de componentes a ser montado. Após a montagem dos *n* primeiros componentes, caso *n* seja menor do que o número de componentes total a ser montado, define-se em seguida uma *seqüência 2* de ventosas necessárias para montar os *m* seguintes componentes da seqüência de montagem. E assim sucessivamente. A seguir ilustra-se esse processo de definição dos conjuntos de ventosas para montagem dos componentes.

Considere-se a hipótese de que uma PCI seja constituída por 16 componentes: c1, c2, c3, c4, c5, c6, c7, c8, c9, c10, c11, c12, c13, c14, c15 e c16. Vamos supor também que esses 16 componentes sejam montados com as seguintes ventosas: h1, h2, h1, h3, h2, h3, h4, h1, h2, h3, h5, h6, h1, h2, h2, h1. A partir dessa lista de ventosas conclui-se que os componentes de c1 a c10 podem ser montados com a *seqüência 1* de ventosas dado por: *seqüência* 1 = {h1, h2, h3 e h4}. Nesse caso, o valor de *n* é igual a 10. Na seqüência de montagem observe-se que os componentes de c11 a c16 podem ser montados com a *seqüência 2* de ventosas dados por: *seqüência 2* = {h5, h6, h1, h2}. Nesse caso o valor de *m* é igual 6. Assim, para a montagem de todos os componentes teríamos que ter apenas uma troca de ventosas.

A montagem dos 10 primeiros componentes é realizada através de 4 ciclos de montagem. No primeiro ciclo serão montados os componentes c1 e c2. No segundo ciclo serão montados os componentes c3, c4 e c5. No terceiro ciclo serão montados os componentes c6, c7, c8 e c9. No quarto ciclo será montado o componente c10.

Para a montagem dos 6 últimos componentes utilizam-se apenas dois ciclos de montagem. No primeiro ciclo são montados os componentes c11, c12, c13 e c14. No segundo ciclo são montados os componentes c15 e c16.

Enfatiza-se novamente que a otimização relacionada à troca de *ventosas*, foi realizada visando maximizar o número de trocas de *ventosas* durante a mesma visita ao ATC, na crença de que, do ponto de vista da otimização do tempo total, uma visita ao ATC é bastante dispendiosa. A troca de várias ventosas numa mesma visita ao ATC merece, no entanto, algumas considerações adicionais. Do ponto de vista de tempo, a troca de várias ventosas é eficiente quando existem tantos espaços vazios adjacentes quanto forem o número de ventosas a serem trocadas e ainda quando as novas *ventosas* a serem capturadas encontram-se adjacentes, e na mesma ordem das cabeças que irão capturá-las. Contudo, apesar de até ser possível realizar tal distribuição, na prática trata-se de uma formulação/processo complexo para ser inserido em um algoritmo como o subida de encosta. Ter-se-ia que simular todas as possibilidades de distribuição das *ventosas* sobre as cavidades do ATC, levando-se em consideração inclusive, a quantidade de *ventosas* por tipo. Ou seja, um trabalho realmente complexo. Diante disso, e baseado na observação prática, percebeu-se que uma distribuição inicial de ventosas no ATC conforme a ilustrada na figura 3.5 (a seguir), atende a todas as necessidades dos diversos modelos de PCI comerciais que se pôde avaliar. Nessa figura, *n* refere-se a um determinado tipo de *ventosa* e cada quadrinho representa uma cavidade do ATC. Assim, inicialmente, nas quatro primeiras cavidades do ATC temos um mesmo tipo de ventosa. Nas quatro cavidades seguintes do ATC temos mais quatro ventosas de um outro tipo, e assim sucessivamente. Essa distribuição é adequada para as seqüências montadas pelo otimizador HLC do ambiente em que se trabalhou (fábrica), bem como serviu para a aplicação dos métodos 1 e 2.

n1	n1	n1	n1	n2	n2	n2	N2	n3	n3	n3	n3	n4	n4	n4	n4	n5	n5	n5	n5

Figura 3.5 – Distribuição de *ventosas* sobre o ATC;

A maneira como é realizado o cálculo do tempo de montagem dos componentes, definida no bloco *Cálculo do tempo t de montagem* será abordada mais adiante, no item 3.2.

A seguir abordar-se-á com mais detalhes as operações realizadas no bloco *gerar vizinho do estado n*, mostrado no fluxograma da figura 3.4.

A vizinhança de um estado é definida a partir de uma permutação aleatória entre dois componentes na seqüência de montagem e por uma permutação aleatória entre dois componentes na seqüência de alimentação. O algoritmo permite na definição de um vizinho, especificar a quantidade de permutações e se estas permutações devem, obrigatoriamente, envolver um componente do primeiro ciclo. Portanto, optou-se por, na definição de um vizinho, fixar o número de permutações em apenas uma. O motivo para isso baseia-se na definição de vizinho como sendo um estado com uma configuração de componentes bem próxima a configuração do estado presente. Nas figuras 3.6 e 3.7 ilustra-se o processo de geração de um estado vizinho.

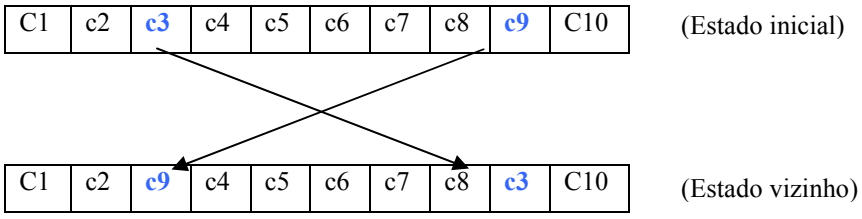

Figura 3.6 – Troca aleatória entre dois componentes na seqüência de montagem para formação de um estado vizinho

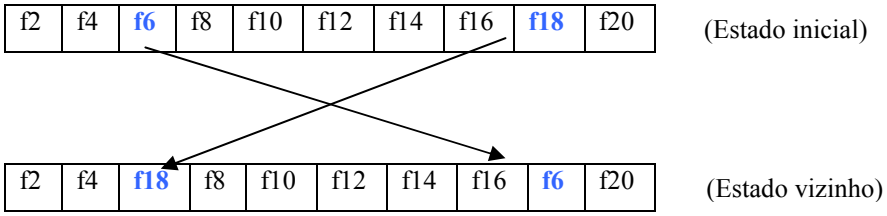

Figura 3.7 – Troca aleatória entre dois componentes nos alimentadores para formação de um estado vizinho

Após a geração de um estado vizinho, todo o procedimento anterior deve ser repetido, calculando-se ao final o valor da função objetivo que corresponde a distância total ou seu equivalente em tempo de montagem para este estado. Caso este tempo seja menor que o calculado para o estado anterior, o algoritmo deve guardar esse novo tempo, bem como o estado correspondente ao mesmo em substituição aos valores anteriormente armazenados. Aqui se encontra uma das características de busca do método subida de encosta, ou seja, procurar dentro do espaço modelado, uma situação onde a função objetivo tenha um valor menor do que o valor previamente armazenado. Trata-se de uma busca aleatória de mínimo local ou global.

A avaliação de estados vizinhos continuará até que se atinja uma quantidade de vizinhos que pode ser configurada no algoritmo (*Lim1*), ou seja, definiu-se um critério de parada (saída) do laço interno do algoritmo. A estratégia adotada foi variar o critério de parada e verificar suas conseqüências na qualidade dos resultados finais obtidos. Desta forma, começou-se com valores baixos do tipo 100 ou 200 vizinhos estendendo-se para quantidades mais altas do tipo 2000 ou 4000 vizinhos. Vide capítulo dos Resultados para avaliação dos tempos de processamento e de montagens obtidos, tanto nas simulações quanto na prática.

Ainda seguindo o cerne do algoritmo de subida de encosta, após a saída do laço interno, deve-se realizar um salto aleatório do estado corrente, com a intenção de "fugir" de um possível mínimo local (mínimo, pois se trata de minimizar o tempo ou distância). Após a geração de um novo estado inicial (reinicio aleatório), entra-se

novamente no laço interno, comparando-se os tempos obtidos pelos vizinhos desta nova região, começando com uma comparação com melhor estado obtido na região anterior. O reinício aleatório constitui o laço externo, para o qual também se definiu um critério de parada análogo ao usado para o laço interno (*Lim2*).

3.1.2 Método heurístico 2

A seguir mostra-se uma lista das etapas do segundo método proposto para aplicação do algoritmo de subida de encosta com reinício aleatório voltado a resolver o problema de inserção de componentes.

Após definir se haverá ou não repetição de ventosas iguais no mesmo ciclo, e admitindo-se que um ciclo seja composto por no máximo quatro componentes, Deve-se:

1) Distribuir, aleatoriamente, os componentes a serem montados com suas respectivas *ventosas* associadas; Distribuir de forma aleatória, os *alimentadores* de componentes sobre os *escaninhos* disponíveis no magazine. Isso corresponde a formação de um estado aleatório.

A figura 3.8 ilustra uma distribuição de *ventosas* conforme a seqüência aleatória de montagem de *n* componentes. Neste exemplo, os ciclos estão separados pelas cores e não há repetição de *ventosas* no mesmo ciclo, conforme pode ser observado na segunda linha da figura. Esta linha corresponde a uma distribuição aleatória de *ventosas*. Onde a notação *hi* foi substituída por *50i*, com *i=1,2,3,4,...*

c1	c2	c3	c4	c5	C6	C7	c8	c9	c10	c11	c12	c13	c14	c15	c16	c17	c18	c19	c20
501	502	504	503	504	501	503	502	502	505	506	501	504	505	502	503	501	502	503	504

1º ciclo **2º ciclo** **3º ciclo** **4º ciclo** **5º ciclo**

Figura 3.8 – 1ª linha: Componentes conforme seqüência de montagem aleatória; 2ª linha: *Ventosas* associadas aos componentes.

2) Otimizar as trocas de *ventosas* conforme distribuição do tipo observada na 2ª linha da figura 3.8, levando-se em conta as seguintes hipóteses:

Hipótese1: Se até quatro *ventosas* forem necessárias para montar todos os componentes, então o nº. de visitas ao ATC para formação de novas cargas de *ventosas* (trocas) será zero no caso de não poder repetir *ventosas* na mesma carga ou ciclo. Porém poderá haver trocas caso haja repetição de *ventosas* no mesmo ciclo.

Hipótese2: Se até oito *ventosas* forem necessárias e não houver repetição de *ventosas* na 1ª carga (*ventosas* do 1º ciclo), então o nº. ótimo de visitas ao ATC para formação de novas cargas de *ventosas,* será de apenas uma. Porém, se houver repetição não há garantias disso.

A figura 3.9 representa uma situação onde cinco *ventosas* são necessárias para a montagem de todos os componentes. Neste caso seria necessário a formação de uma segunda carga de *ventosas* onde haveria a troca da *ventosa* 501 pertencente à primeira carga formada pela *ventosa* 505, uma vez que existem componentes que só poderiam ser montados com o auxílio desta *ventosa*.

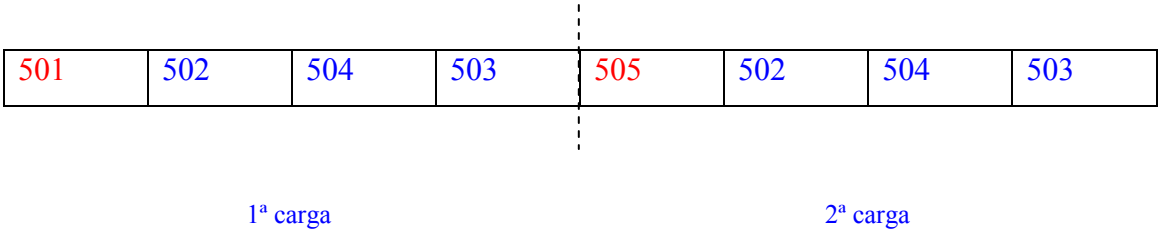

Figura 3.9 – Formação de cargas de *ventosas* sem repetição e apenas uma troca da *ventosa* 501 para a ventosa 505 na formação da 2ª carga; A linha pontilhada representa esta troca de carga no ATC. Há cinco *ventosas* no total.

Hipótese3: Se até doze *ventosas* forem necessárias, então o nº. ótimo de formação de novas cargas, será de apenas duas (sem repetição). Ou um número a ser determinado (com repetição).

A figura 3.10 representa a situação onde dez *ventosas* são necessárias para a montagem de todos os componentes. Desta forma, todos os componentes que podem ser montados com a primeira carga serão montados. A seguir, será realizada a primeira visita ao ATC para a troca das *ventosas* que comporão a segunda carga e conseqüente montagem dos componentes inerentes. Finalmente, as duas últimas *ventosas* que faltaram são carregadas na terceira carga de *ventosas*, a qual corresponde a uma segunda visita ao ATC para troca das *ventosas* 506 e 507 pelas ventosas 509 e 510 respectivamente.

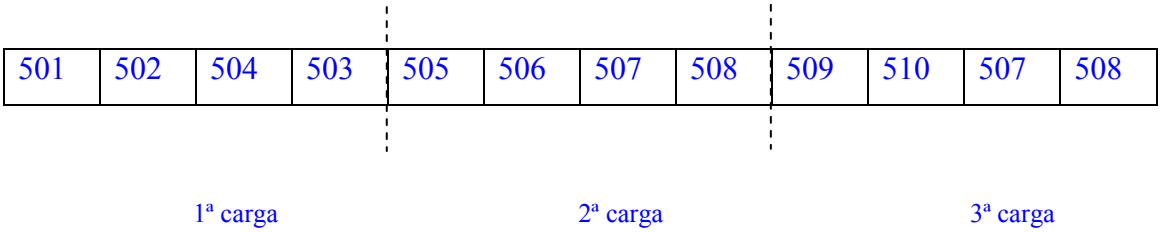

Figura 3.10 – Formação de cargas de *ventosas* sem repetição e com duas novas cargas formadas; A linhas pontilhadas representam o ponto de troca de carga no ATC; Dez *ventosas* no total.

Hipótese4: Se até N *ventosas* forem necessárias, então o número ideal de visitas ao ATC para formação de novas cargas de *ventosas*, será a parte inteira da divisão de N por quatro (nº. de componentes de um ciclo para uma máquina com quatro cabeças e N>4). Ou seja, o nº. de novas cargas formadas seria igual ao quociente desta conta no caso de não haver *ventosas* repetidas no mesmo ciclo. Caso haja tal repetição, então este valor poderá variar.

3) Medir as distâncias ou tempo de inserção para montar a seqüência formada e otimizada;

4) Comparar o valor medido com melhor valor encontrado até então e seu respectivo estado (seqüência de montagem e de alimentação);

5) Gerar vizinho do estado avaliado;

6) Voltar ao passo 2 enquanto o critério de parada não for satisfeito. Exemplo: nº. máximo de vizinhos não foi atingido;

7) Voltar ao passo 1 enquanto o critério de parada não for satisfeito. Exemplo: n°. máximo de estados aleatórios não foi atingido.

8) Plotar gráficos com as soluções (baseadas no cálculo da função objetivo) versus a época do algoritmo;

Na figura 3.11, mostra-se um fluxograma do segundo método. O valor de *to* refere-se ao tempo da melhor montagem. Inicialmente o mesmo assume um valor muito grande, representado pelo símbolo ∞. Os parâmetros *n* e *m* controlam o número de estados gerados e o número de vizinhos gerados para um estado. Os parâmetros *Lim1* e *Lim2* estabelecem o número máximo de vizinhos gerados para um estado e o número máximo de estados gerados.

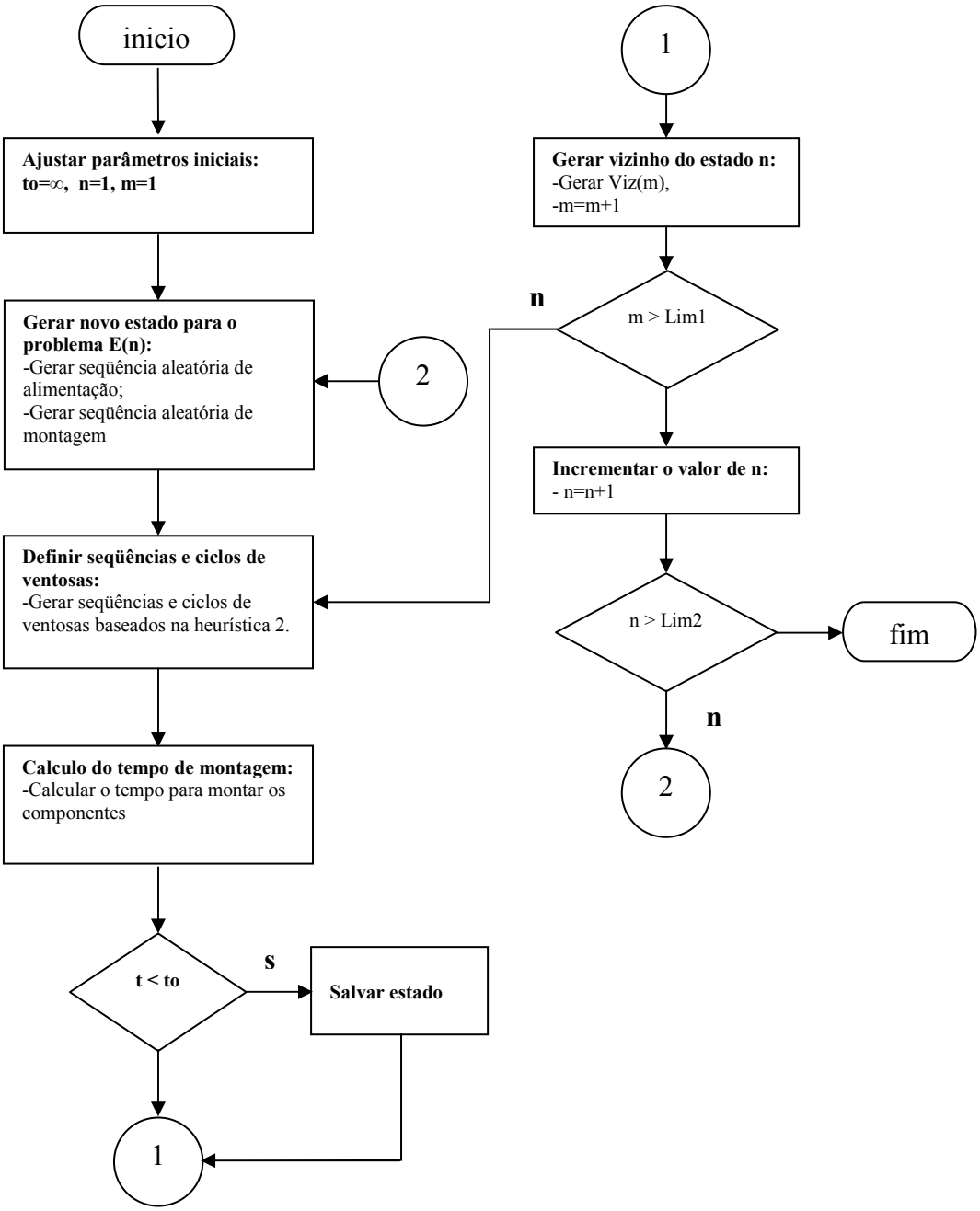

Figura 3.11: Fluxograma para a heurística 2

O processo de geração de estados vizinhos, após a definição um estado aleatório inicial, seguiu os mesmos procedimentos estabelecidos para a proposta 1. Contudo, a heurística usada para a formação dos ciclos em função da otimização da troca de *ventosas*, constitui um dos fatores de diferença entre as duas propostas. Outro fator, conforme mencionado no início, diz respeito à possibilidade de repetirem-se *ventosas* em um mesmo ciclo.

Depois de estabelecida uma seqüência de montagem inicial, com respectiva distribuição de *ventosas* e *alimentadores* necessários, o algoritmo passa a resolver o problema de minimização de troca de *ventosas* da forma seguinte:

Identifica-se a seqüência de montagem dos componentes do primeiro ciclo (normalmente constituído por quatro elementos no caso de uma máquina com até quatro cabeças de inserção). A partir daí, será realizada uma busca dentre todos os demais componentes da seqüência, para formar ciclos consecutivos, onde os componentes sejam inseridos com as mesmas *ventosas* presentes no primeiro ciclo. As figuras 3.12a e 3.12b ilustram esse procedimento.

c1	c2	c3	c4	c5	c6	c7	c8	c9	c10	c11	c12	c13	c14	c15	c16	c17	c18	c19	c20
501	502	501	503	504	501	503	504	502	502	501	501	504	504	502	503	501	502	503	504

1º ciclo **2º ciclo** **3º ciclo** **4º ciclo** **5º ciclo**

Figura 3.12a – Seqüência fictícia de montagem com 20 componentes e 5 ciclos separados pela cor; 1ª linha componentes; 2ª linha *ventosas*.

c1	c2	c3	c4	c6	c9	c11	c7	c12	c10	c17	c16	x	c15	x	c19	x	c18	x	x
501	502	501	503	501	502	501	503	501	502	501	503	x	502	x	503	x	502	x	x

1º ciclo **2º ciclo** **3º ciclo** **4º ciclo** **5º ciclo**

Figura 3.12b – Rearranjo a partir do segundo ciclo tomando-se o 1º ciclo como referência; As posições com *x* formarão ciclos extras após a captura da *ventosa* 504.

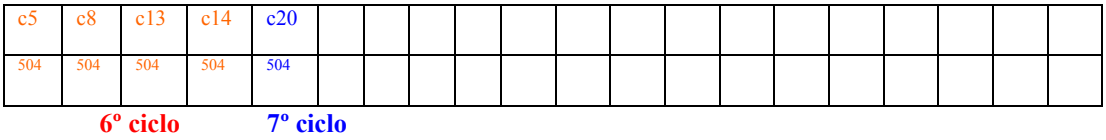

Figura 3.12c – Opção de troca "simultânea" de quatro *ventosas*: 6º ciclo trocando-se 501->504; 502->504; 501->504; 503->504; 7º ciclo não há troca.

Deve-se atentar para o fato de haver possibilidade de formação de ciclos onde menos de quatro (e até mesmo apenas um) componentes sejam montados, como ilustrado na figura 3.12b, onde o ciclo mais a direita possui somente um componente. É claro que o máximo de quatro veio da suposição de tratar-se de uma máquina com quatro cabeças. A principal razão para permitir-se um ciclo com um número menor de componentes em relação ao número de cabeças, reside no fato de que é preferível adiar a montagem de um componente, mesmo que isso represente um aumento no número de ciclos de montagem, do que realizar uma visita ao ATC para trocar uma *ventosa*. Porém, há um limite para tal procedimento, pois a partir de um determinado número de espaços vazios em ciclos de montagem, seria melhor realizar uma visita ao ATC para troca de *ventosas*. Basta ponderar se a quantidade de ciclos extras criados gerará um tempo de montagem maior do que o gasto ao visitar-se o ATC para se trocar uma única *ventosa*. Contudo, a observação demonstrou que para seqüências de montagens relativamente pequenas, a quantidade de ciclos extras gerados, em geral, não atinge este valor crítico. Esta foi à razão pela qual não se observa nenhum controle deste tipo no fluxograma da figura 3.11. Posteriormente, entretanto, uma alteração poderá ser facilmente realizada para comportar esta particularidade.

O restante dos componentes que faltaram ser computados formou os ciclos seis e sete ilustrados na figura 3.12c. O ciclo seis identifica a troca simultânea de quatro *ventosas*, ou seja, as *ventosas* 501, 502, 501 e 503, que respectivamente encontravam-se nas extremidades das cabeças h1, h2, h3 e h4, foram (todas) substituídas por *ventosas* do tipo 504. Já o sétimo ciclo, formado apenas pelo componente *c20*, aproveitou o fato de também necessitar do mesmo tipo de *ventosa* (504). Portanto, não houve nova troca de *ventosas*. Este tipo de troca envolvendo quatro trocas de *ventosas* em uma única visita ao dispositivo de ATC, foi baseado na tendência em aproveitar-se a visita para

trocar o máximo de ventosas que fossem necessárias. Daí trocar-se quatro *ventosas* nesta visita.

Esta proposta foi avaliada tanto para um critério onde os ciclos não permitam a repetição de *ventosas* entre as cabeças (diferentemente do que está ilustrado acima), quanto para um outro critério onde se permitia esta repetição de *ventosas* (conforme ilustrado acima).

Os critérios de parada dos laços interno e externo foram semelhantes aos usados na proposta 1. Sendo assim, definiu-se uma quantidade parametrizada para geração e avaliação dos tempos de montagem das seqüências dos vizinhos do estado inicial (laço interno), bem como, um outro parâmetro para a geração de estados aleatórios iniciais (laço externo).

3.2 Modelagem matemática para a função objetivo

Trata-se de uma modelagem aplicada a todos os métodos heurísticos que foram avaliados (propostas). Aproveitou-se a notação matemática que pode ser encontrada em alguns trabalhos realizados conforme a lista de referencias encontrada no final deste texto. Na verdade, existem diferenças em função do contexto de cada trabalho, mas o cerne para o cálculo das funções objetivo é sempre baseado nas distancias percorridas. Um destes trabalhos [Ayob & Kendall, 2003] apresenta uma notação que foi parcialmente usada neste texto. Porém houve diferenças quanto à forma de cálculo das distâncias percorridas durante os eventos de captura e montagem de componentes. Enquanto aquele usou distancias Chebchev, a modelagem presente usou distâncias Euclidianas. Ademais, a modelagem atual não restringiu a ocupação dos *escaninhos* a um único tipo de componente, ou seja, pôde-se ocupar mais de um *escaninho* com o mesmo tipo de componente [3.2.1.6 - nota2]. Outra diferença a favor da implementação metodológica ora examinada, diz respeito à não restrição de tipos de *ventosas* distintos em um mesmo ciclo, ou seja, podem-se ocupar as cabeças de inserção de componentes, havendo repetição do tipo de *ventosa* em duas ou mais cabeças (ou cabeçotes) [3.2.1.6 - nota3].

3.2.1 Etapas que compõem um ciclo de montagem

Antes de expor a notação e respectiva formulação, cabe uma explicação sucinta a respeito das etapas que compõem a medição da distância ou seu equivalente tempo de montagem durante um ciclo.

Definiu-se como ciclo de montagem a união dos eventos necessários para preencher as cabeças (no caso presente em número de quatro) com os componentes capturados nos *escaninhos-alimentadores* e montá-los em posições específicas sobre a PCI. A soma dos eventos de todos os ciclos determina a montagem de todos os componentes sobre uma PCI. A função objetivo ou função de custo vista em 3.2.2 como CT – Ciclo Total é quem mensura o tempo gasto para a montagem completa.

Desta forma, se a máquina de inserção possuir um dispositivo de montagem com no máximo 4 cabeças para captura de componentes e se a PCI possuir 40 componentes a serem montados, então, uma forma de calcular o quantidade de ciclos seria dividir o número de componentes a serem montados sobre a PCI pelo número máximo de cabeças de montagem da máquina, ou seja, chagar-se-ia a quantidade de 10 como sendo o número de ciclos necessários. Entretanto, como se está tratando de um método de otimização, esta estimativa serviria apenas como um ponto de partida conforme fica claro no decorrer da dissertação.

3.2.1.1 Distância percorrida para capturar todos os componentes dos *alimentadores* em um ciclo

Com o dispositivo de múltiplas cabeças (vide figuras 3.13a e 3.13b) já posicionado sobre o magazine onde se encontram os *escaninhos* com os *alimentadores* de componentes, inicia-se a mensuração das distancias Euclidianas percorridas baseado nas coordenadas de cada *alimentador*. Como se trata aqui de uma máquina com 4 cabeças, h1, h2, h3 e h4, somar-se-á no máximo 3 segmentos. Por exemplo, h1-h2; h2-h3; h3-h4; ou h1-h4;h4-h2;h2-h3, etc

Onde:

h1-h2 representa uma seqüência onde o 1º componente será capturado por h1 nas coordenadas *x1,y1* onde está posicionado o *alimentador* que contem aquele componente. Deslocando-se logo após, para a posição de coordenada *x2,y2*, onde se encontra o *alimentador* com o componente a ser capturado com a cabeça h2.

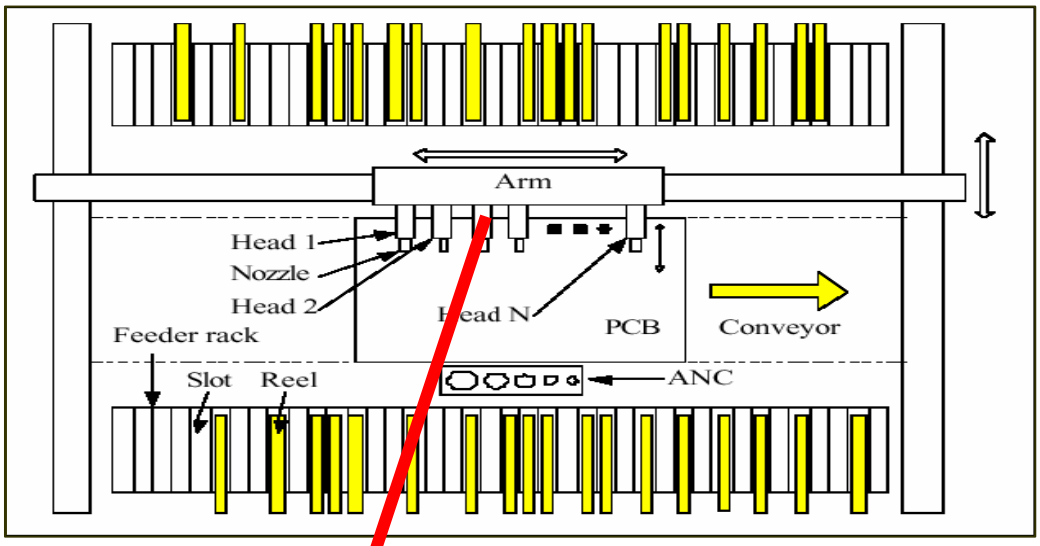

Figura 3.13a – Destaque do deslocamento do repouso do bloco de cabeças (head) até o magazine de escaninhos frontal no ciclo inicial.

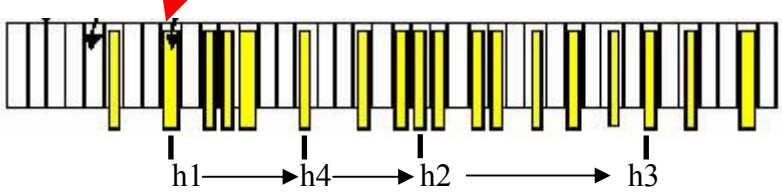

Figura 3.13b – Deslocamento sobre o magazine de escaninhos

3.2.1.2 Distância percorrida para transportar os componentes capturados até a plataforma da PCI

Após capturar o último componente de um ciclo, o conjunto com as cabeças carregadas deve deslocar-se para a plataforma fixa onde se encontra a PCI. Desta forma, a distancia percorrida será medida a partir das coordenadas do último componente capturado (*alimentador* que contem este componente) até as coordenadas do primeiro componente montado sobre a PCI no ciclo corrente.

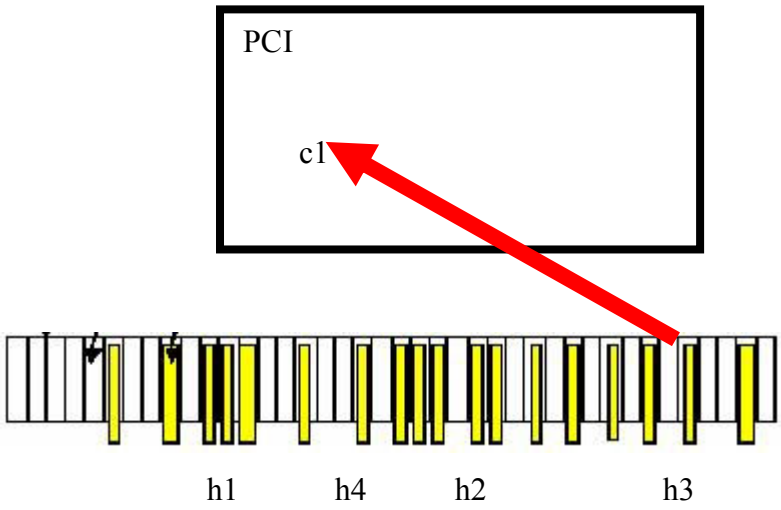

Figura 3.14 – Deslocamento do bloco da posição do último componente capturado até o primeiro ser montado sobre a PCI

3.2.1.3 Distância percorrida para montar os componentes capturados sobre a PCI

A seqüência de montagem dos componentes sobre a PCI determinará qual cabeça inserirá o componente que capturou por primeiro, por segundo, etc.

Novamente teremos no máximo 3 segmentos para montar no máximo 4 componentes por ciclo. As distâncias Euclidianas destes segmentos serão calculadas e somadas em função das coordenadas sobre a PCI onde deverão ser inseridos os componentes na seqüência determinada.

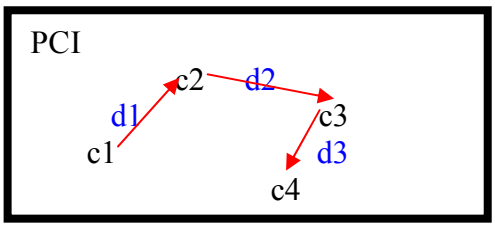

Figura 3.15 – Seqüência de montagem sobre a PCI com três segmentos d1, d2 e d3

3.2.1.4 Distância percorrida para transportar o conjunto de cabeças até os *alimentadores* (início de um novo ciclo)

Após montar o último componente, o conjunto de múltiplas cabeças deverá retornar ao magazine para iniciar novo ciclo de captura e montagem. Então as distancia percorrida pelo dispositivo com as cabeças deverá ser calculada entre as coordenadas do último componente montado sobre a PCI e as coordenadas do primeiro componente a ser capturado, o qual se encontra em um determinado *alimentador*.

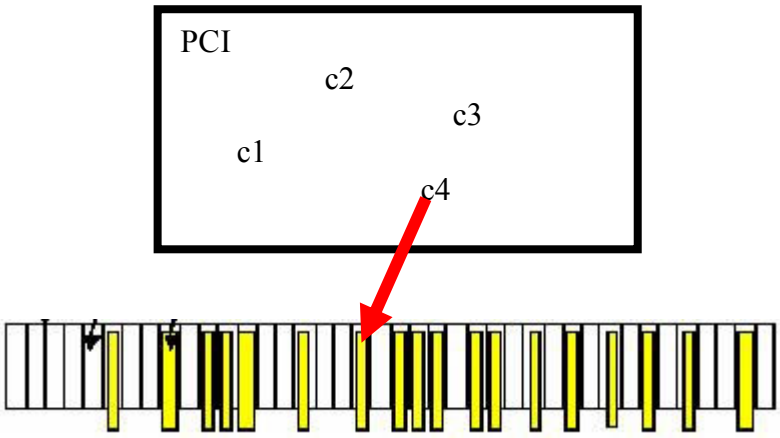

Figura 3.16 – Deslocamento do último componente montado no ciclo atual até o primeiro montado no ciclo seguinte

3.2.1.5 Distância envolvendo a troca de *ventosas*

Conforme esclarecido na introdução, as *ventosas* são dispositivos encaixados nas extremidades das cabeças de captura e montagem. Sua função é permitir a captura de um determinado tipo de componente baseado em seu peso e tamanho, ou seja, para cada tipo ou grupo de tipos de componentes existem tipos apropriados de *ventosas*. Assim sendo, pode haver a necessidade de trocar uma ou várias *ventosas* durante os eventos cíclicos. Tal necessidade vai depender da seqüência de montagem que foi estabelecida. Sabe-se que não é desejável haver qualquer tipo de troca de *ventosa* durante o processo de montagem, uma vez que ele é muito lento em comparação com eventos como a captura ou a montagem de um componente. A troca de *ventosa* tem grande influencia na otimização do tempo de montagem!

O ATC é um dispositivo que fica próximo à plataforma da PCI (vide figura 3.13a - ANC). Sobre este dispositivo é que se encontram as *ventosas* a serem trocadas. Trata-se de um dispositivo fixo, ou seja, suas coordenadas independem da seqüência de montagem ou da PCI a ser montada. Na verdade, há uma coordenada para cada *ventosa* presente neste dispositivo. No caso da máquina JUKI-2030, há possibilidade encaixe de até 21 *ventosas* dispostas linearmente.

Se houver a necessidade de trocar pelo menos uma *ventosa* na seqüência final de montagem, a distância Euclidiana, **d1**, calculada entre as coordenadas do último componente montado e as coordenadas do ATC também deve ser computada. Então, após a troca da *ventosa,* o bloco com as cabeças deve se deslocar até o magazine para a captura de novos componentes (novo ciclo) e, assim como antes, a distância Euclidiana, **d2**, calculada entre as coordenadas do ATC e as coordenadas do 1º componente a ser capturado no próximo ciclo, deve ser computada.

Além destes dois segmentos de distancias, **d1 e d2**, o valor em unidade de tempo correspondente ao processo de reposição da *ventosa* anterior sobre o ATC e captura da nova *ventosa*, **tanc,** também deve ser mensurado e somado ao tempo total de montagem da PCI. A figura 3.17 ilustra o que foi dito.

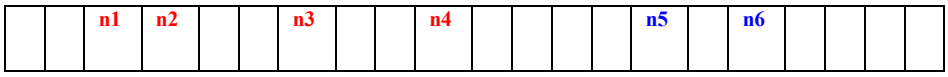

Figura 3.17a – ATC com capacidade para até vinte e uma ventosas. Em vermelho representam-se as *ventosas* que já foram capturadas; Em azul as *ventosas* atualmente presentes no ATC.

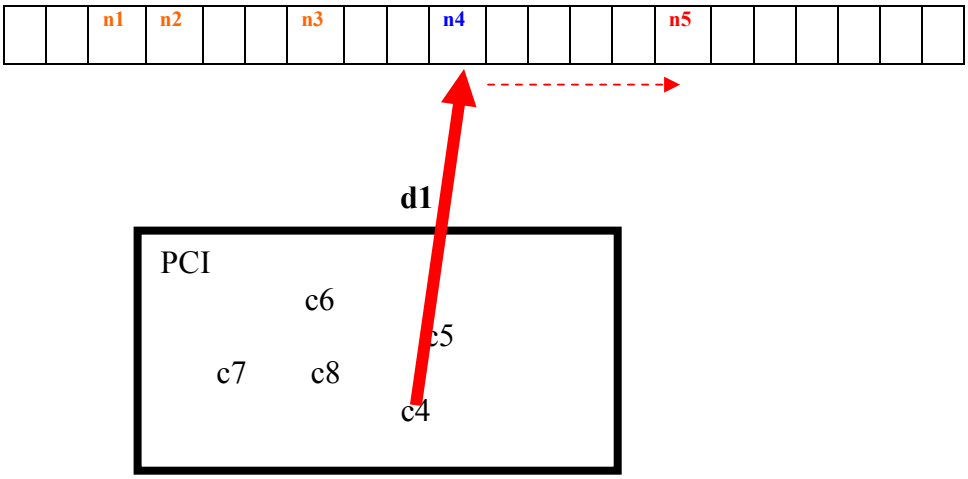

Figura 3.17b – Deslocamento - d1 até o ATC. Após montar c4, devolve-se o n4 que estava em uma das cabeças; Depois, captura-se a nova *ventosa* – n5 no lugar de n4.

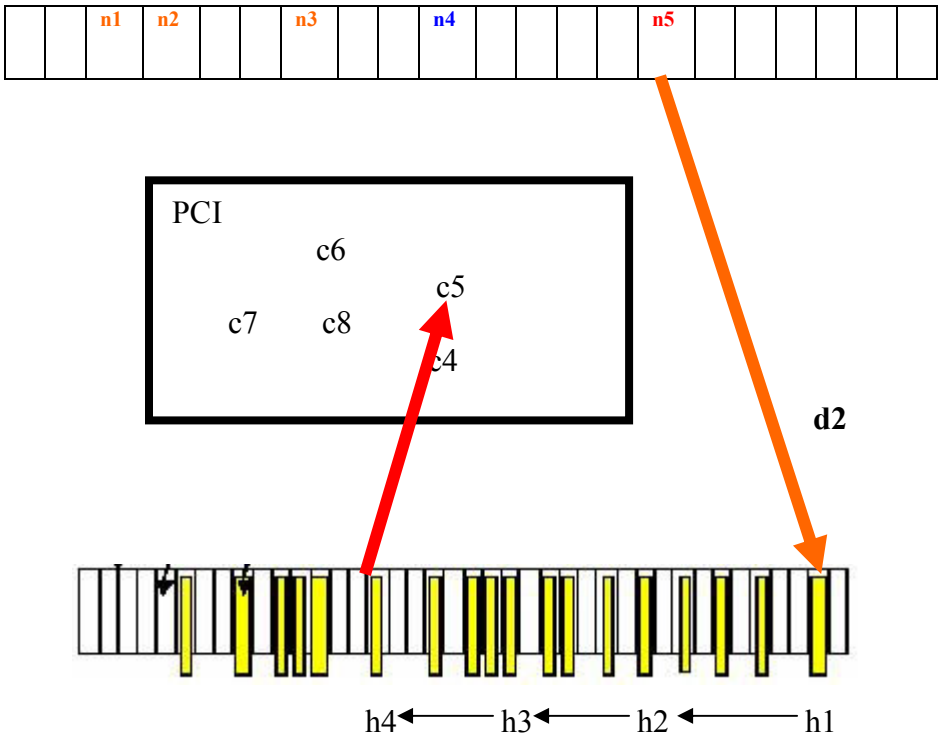

Figura 3.18 – Deslocamento – d2 do ATC até o magazine para captura de componentes do ciclo seguinte. Após a captura do último componente, inicia-se a seqüência de montagem começando pelo c5.

3.2.1.6 Definições e notações matemáticas para a função objetivo

a) <u>Definições:</u>

CT: O ciclo de tempo total para capturar e montar todos os componentes sobre uma PCI;

N: O número de pontos de montagem sobre uma PCI;

Q: O número total de pontos de montagem disponíveis para a seqüência; onde $Q \leq N$;

K: O número de tipos de componentes (cada alimentador-escaninho mantém inúmeras cópias de um tipo de componente);

G: O número de *ventosas* para cabeças (cabeças);

T: O número total de ciclos;

M: O número total de alimentadores-escaninhos, onde $K \leqslant M$;

V_r : A velocidade média do braço do robô;

T_p : O tempo para capturar um componente;

T_i : O tempo para montar um componente;

F: A distância entre alimentadores-escaninhos adjacentes - *gap*;

L: A distância entre ventosas adjacentes - *gap*;

Usando j para representar o ciclo de montagem atual com $1 \le j \le K$ e usando k para representar a seqüência atual dentro de um determinado ciclo j, onde $1 \le k \le G$, temos que:

$c(j,k)_{x,y}$: As coordenadas x,y sobre a PCI que terá um componente na $k^{ésima}$ seqüência do $j^{ésimo}$ ciclo;

$I(j,k)$: O tempo necessário para o braço viajar e montar um componente na $k^{ésima}$ seqüência do $j^{ésimo}$ ciclo;

$P(j,k)$: O tempo necessário para o braço viajar e capturar um componente na $k^{ésima}$ seqüência do $j^{ésimo}$ ciclo;

$h_p(j,k)$: A ventosa usada para **capturar** um componente na $k^{ésima}$ seqüência de captura do $j^{ésimo}$ ciclo de pick&place;

$h_b(j,k)$: A ventosa usada para **montar** um componente na $k^{ésima}$ seqüência de captura do $j^{ésimo}$ ciclo;

$s(i)$: A distancia do $i^{ésimo}$ escaninho referente a origem dos alimentadores-escaninhos, $s(0)$, onde $s(0) = 0; i < k$;

$r(j,k)$: A distância do escaninho para a $k^{ésima}$ seqüência de captura do $j^{ésimo}$ ciclo;

$d(j,k)$: É a distancia máxima entre $\{|d(j,k)x|, |d(j,k)y|\}$, onde x e y são as distancias X, Y para o braço do robô **montar** um componente na $k^{ésima}$ seqüência de montagem do $j^{ésimo}$ ciclo, sendo esta distância medida como uma distância Chebychev;

$m(j,k)$: A distância máxima entre $\{|m(j,k)x|, |m(j,k)y|\}$, onde x e y são as distâncias X, Y para o braço do robô **capturar** um componente na $k^{ésima}$ seqüência de captura do $j^{ésimo}$ ciclo, sendo esta distância medida como uma distância Chebychev;

b) <u>Formulação matemática</u>

$$CT = \min \sum_{j=1}^{T} \left[\sum_{k=1}^{G} P(j,k) + \sum_{k=1}^{G} I(j,k) \right] \quad \text{(a)}$$

onde,

$P(j,k) = (m(j,k)/V_r + T_p) * z(j,k);$

$I(j,k) = (d(j,k)/V_r + T_i) * z(j,k);$

sendo que,

$z(j,k)$ é a ventosa usada para capturar o componente na $k^{ésima}$ seqüência do $j^{ésimo}$ ciclo, e

$z(j,k) = 1$; Se há *ventosa* para capturar ou montar um componente na $k^{ésima}$ seqüência do $j^{ésimo}$ ciclo;

$z(j,k) = 0$; Em qualquer caso contrário;

Além disso:

$\sum_{k=0}^{G-1} z(j,k) \le G;$ Significa que podemos ter ciclos com no máximo G componentes, e

T = Q/G se Q%G=0;

T = 1+Q/G se Q%G≠0;

$m(j,k) = \max\{m(j,k)_x, m(j,k)_y\}$; Distância Chebychev

$d(j,k) = \max\{d(j,k)_x, d(j,k)_y\}$; Distância Chebychev

À distância Chebychev poderia ser considerada em função dos motores de movimento do braço nas direções x e y serem concorrentes;

Nota1:

Entretanto, tanto na simulação, quanto na experiência prática, optou-se, neste trabalho, pela medição da distância em termos da formulação Euclidiana conforme expressões abaixo:

$m(j,k) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$; (de forma genérica)

$d(j,k) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$; (de forma genérica)

Onde *x e y* são as coordenadas da PCI com translado da origem da PCI para a origem da máquina de inserção automática. Pois, as coordenadas da PCI são diferentes das coordenadas de posição dos braços robóticos e, obviamente, necessita-se de um sistema de coordenadas com mesma referência.

Continuando:

$\sum_{j=0}^{K-1} g_{ij} \leq 1, \forall i;$

Nota2:

A expressão acima se restringe a 1 (um), quando pode haver no máximo um tipo de componente por *escaninho*. Porém, no modelo criado isso não se faz necessário, ou seja, um determinado tipo de componente pode ser alocado em mais de um *escaninho*. Onde *i* representa os tipos de componentes.

$s(i) = \sum_{j=0}^{K-1} [j * F * g_{ij}]$; Assumindo que os *alimentadores* são encaixados em *escaninhos* adjacentes.

$\begin{cases} b_i(j,k) = 1 \text{ ; Se o componente do tipo i está presente na } k^{ésima} \text{ seqüência de captura do } j^{ésimo} \text{ ciclo.} \\ b_i(j,k) = 0 \text{ ; Em qualquer caso contrário.} \end{cases}$

$r(j,k) = \sum_{i=0}^{K-1} [s(i) * b_i(j,k)]; \ \sum_{i=0}^{K-1} b_i(j,k) \leq 1, \forall j, \forall k \ ;$

$\begin{cases} g_{ij}(j,k) = 1 \text{ ; Se o componente do tipo } i \text{ está alocado no } alimentador \text{ que está no } escaninho\ j. \\ g_{ij}(j,k) = 0 \text{ ; Em qualquer caso contrário.} \end{cases}$

$h_p(j,k) = \{0,1,2,...,(G-1)\}$; ou $\{1,2,...,(G)\}$; (vale para o restante)

desde que:

$h_p(j,k) \neq h_p(j,m)$ e $k = \{0,1,2,...(G-1)\}; \ j = \{0,1,2,...(T-1)\}.$

$h_b(j,k) = \{0,1,2,...,(G-1)\};$

$h_b(j,k) = \{0,1,2,...,(G-1)\};$

desde que:

$h_b(j,k) \neq h_b(j,m)$ e $k = \{0,1,2,...(G-1)\}; \ j = \{0,1,2,...(T-1)\}.$

Nota3:

No caso presente, porém, não há esta necessidade, ou seja, pode haver *ventosas* iguais no mesmo ciclo *j;*

Finalmente, temos que:

$$\begin{cases} m(j,k)_x = r(j,k) - h_p(j,k) * L \text{ , se } j = k = 0 \text{ ; (no início, 1º ciclo, 1ª seqüência)} \\ m(j,k)_x = \left[r(j,k) - h_p(j,k) * L\right] - \left[c(j-1,G-1)_x - h_b(j-1,G-1) * L\right], \\ \text{se } j > 0, k = 0 \text{ ;(1ª seqüência de todos os ciclos exceto o 1º ciclo)} \\ m(j,k)_x = \left[r(j,k) - r(j,k-1)\right] - \left[(h_p(j,k) - h_p(j,k-1)) * L\right], \\ \text{se } k > 0 \text{ ; (restante das seqüências dos demais ciclos exceto o 1º ciclo)} \end{cases}$$

Onde:

a) *(j-1)* é o ciclo anterior; *(G-1)* é a última seqüência no ciclo j; *(k-1)* é a seqüência anterior no ciclo *j*.

b) A expressão $h_p(j,k) * L$ é para compensar a posição da cabeça com a ventosa de captura apropriada para o tipo de componente a capturar, ou seja, o deslocamento entre as cabeças de *1 a G*.

e,

$$\begin{cases} m(j,k)_y = 0 - c(j-1,G-1)_y \text{ , se } j > 0, k = 0 \text{ (deslocamento do escaninho para a PCI).} \\ m(j,k)_y = 0 \text{, em qualquer caso em contrário. (no 1º ciclo é zero porque não vem da PCI).} \end{cases}$$

$$\begin{cases} d(j,k)_x = \left[c(j,k)_x - r(j,G-1) * L\right] - \left[(h_b(j,k) - h_p(j,G-1)) * L\right], \text{ se } k = 0; \\ d(j,k)_x = \left[c(j,k)_x - c(j,k-1)_x\right] - \left[(h_b(j,k) - h_p(j,k-1)) * L\right], \text{ se } k > 0; \end{cases}$$

e,

$$\begin{cases} d(j,k)_y = \left[c(j,k)_y\right], \text{ se } k = 0 \text{ ; (uma vez que a origem dos y reside no carro de escaninhos)} \\ d(j,k)_y = \left[c(j,k)_y - c(j,k-1)_y\right], \text{ se } k > 0; \end{cases}$$

Onde:

a) *c* representa coordenada sobre a PCI e *r* representa coordenada sobre o carro de *escaninhos*;

b) Em todos os casos as distâncias são mensuradas da posição final (para onde o braço se deslocou) subtraída da posição inicial (onde o braço estava).

Por exemplo: $\left[(h_p(j,k) - h_p(j,k-1)\right]$ informa que o conjunto com as cabeças e *ventosas* se deslocou da posição da seqüência *k-1* para a posição da seqüência *k* no ciclo *j*.

c) Os índices *x* e *y* associados distâncias $d(j,k)$ e $m(j,k)$, equivalem as coordenadas x e y das equações Euclidianas (1) e (2) referidas anteriormente.

3.3 Otimização da etapa de captura de componentes

Inicialmente, o tempo, ou seu equivalente em termos de distância percorrida para captura dos componentes *c* nos *escaninhos*, foi mensurado em função da própria seqüência de montagem do ciclo corrente. Então, se a seqüência de montagem fosse *c1-c2-c3-c4*, o tempo de captura seria mensurado partindo-se do *c1* em direção ao *c2*; deste em direção ao *c3*, finalizando no *c4*. Desta maneira, não se avaliaria qualquer outra possibilidade combinatória mesmo que se tratasse de uma combinação culminando em um tempo menor. Por exemplo, se a seqüência *c4-c2-c1-c3* resultasse em um tempo de captura menor, ainda assim seria desprezado. É claro que tal avaliação corresponderia a resolver um problema do tipo caixeiro viajante [Merz & Freisleben, 1997].

Esta análise não foi contemplada nos algoritmos originais, pois se acreditava que o processo de permutações para formação de estados vizinhos ou de reinício aleatório, naturalmente encontraria tais ciclos com menores tempos para captura. Mas, contrariando as expectativas, este fato não ocorreu, como demonstraram os resultados das simulações. Além disso, houve uma discrepância acentuada entre o valor do tempo de montagem gerado pelo simulador e o valor cronometrado após aplicação prática, já considerando-se o fato desses valores não poderem ser iguais devido as imprecisões da própria forma de cronometrar e do valor aproximado que foi empiricamente fixado para velocidade de deslocamento do braço.

Para contornar as distorções oriundas das observações acima, resolveu-se alterar a forma de mensuração do tempo de captura de componentes em um dado ciclo. A nova forma adotada rompeu com a dependência direta que a seqüência de captura tinha com relação à seqüência de montagem. Uma seqüência de montagem dada por *c1-c2-c3-c4*, pode corresponder a uma seqüência de captura dada por *c2-c3-c1-c4*, que pode corresponder a um melhor tempo de captura quando comparado ao tempo de captura da primeira seqüência. Na prática, isso equivaleu a uma convergência "forçada" para a melhor solução de captura na forma de um menor tempo de captura.

Um exemplo prático é ilustrado na tabela 3.1, onde se lista uma seqüência de montagem formada pelos componentes c1, c2, c3 e c4.

Tabela 3.1 – Destaque em azul para seqüência de montagem e respectivas *ventosas*, escaninhos, cabeças; O restante dos parâmetros foram marcados com x (qualquer valor)

Numeração	Coordenada X	Coordenada Y	giro	Código do componente	Cabeça	*Ventosa*	Seqüência de montagem	*Escaninho*
X	X	X	x	x	x	x	X	x
X	X	X	x	x	x	x	X	x
X	X	X	x	c1	h1	501	1	2
X	X	X	x	c2	h2	502	2	40
X	X	X	x	c3	h3	502	3	4
X	X	X	x	c4	h4	503	4	54
X	X	X	x	x	x	x	X	x
X	X	X	x	x	x	x	X	x

Para o caso ilustrado na tabela 3.1, se a captura continuasse a seguir a montagem, ter-se-iam os segmentos na seguinte seqüência: c1-c2; c2-c3; c3-c4. Onde a cabeça h1 capturaria o componente c1 no *escaninho* 2 e logo em seguida partiria para o *escaninho* 40, onde o componente c2 seria capturado pela cabeça h2, por exemplo. Então a distância d1 percorrida na direção *x* seria d1 = (40-2)x17/2=323mm. O valor 17 representa o espaçamento fixo, em mm, existente entre dois *escaninhos* adjacentes. Já a divisão por 2 advém da identificação dos *escaninhos* sempre possuir numerais pares neste caso. Em seguida, a cabeça parte para a captura do componente c3, percorrendo um distância igual a 306mm. Logo após, parte para a captura de c4, percorrendo uma distância igual a 425mm. A distância total percorrida seria de: 323mm+306mm+425mm = 1156mm. Entretanto, se a seqüência de captura fosse c1-c3-c2-c4 teríamos uma distância igual a: 17+306+119= 442mm.

Como se está tratando de uma máquina com no máximo 4 cabeças, existe certa facilidade para se varrer todo o espaço de combinações possíveis para captura em um ciclo com no máximo 4 componentes presentes. Além de se considerar a distância percorrida entre os *escaninhos* (com uma distribuição de *alimentadores* já conhecida), considera-se também o tipo de cabeça que os capturam, perfazendo um total de apenas 4x3x2x1 = 24 combinações possíveis.

No capítulo de resultados mostra-se como houve uma diminuição substancial do tempo de montagem resultante de processo de otimização da captura de componentes.

3.4 Otimização da etapa de troca de ventosas

Durante as explicações sobre os métodos heurísticos, foi dito que a visita ao dispositivo de ATC, onde são realizadas as trocas de *ventosas*, deveriam maximizar as trocas. Em outras palavras, poder-se-ia dizer que as operações de troca de *ventosas* são críticas por serem extremamente lentas, devendo-se evitá-las. Entretanto, havendo a necessidade de trocas, estas deveriam ser realizadas durante a mesma visita ao ATC. Por exemplo, se houvesse a necessidade de trocar três *ventosas*, estas deveriam ser trocadas durante a mesma visita ao ATC.

Assim sendo, os métodos propostos procuraram maximizar as trocas de ventosas em uma visita ao ATC e, no caso de uma máquina com quatro cabeças, a maximização corresponderia a quatro trocas durante uma mesma visita ao ATC, caso houvesse real necessidade para isso.

Porém, baseando-se na observação prática, percebeu-se que o processo se torna extremamente lento, pois se deve devolver uma ventosa ao ATC antes de capturar uma nova. Então, em um dos piores cenários, ter-se-ia que consecutivamente devolver quatro ventosas e da mesma forma, (consecutivamente) capturar outras quatro. Trata-se, portanto de um processo ineficaz. Desta forma, os algoritmos passaram a ser configurados para priorizar não somente quatro, mas também três, duas ou mesmo uma troca em cada visita ao ATC e, para uma mesma seqüência de montagem, avaliar-se qual configuração resultaria em menor tempo de montagem.

Para ilustrar o que foi dito, relembra-se que no item **3.1.2**, priorizou-se a troca de quatro *ventosas* na mesma visita, pois havia a necessidade de trocar-se tal quantidade de *ventosas*. Porém, se a prioridade fosse a troca de até três *ventosas*, ter-se-ia o arranjo mostrado na figura 3.19a; Outra possibilidade seria a de priorizar-se a troca de duas *ventosas* conforme ilustrado na figura 3.19b; Finalmente, se a prioridade fosse à troca de apenas uma *ventosa*, então, ter-se-ía o arranjo da figura 3.19c.

c5	c8	c13	x	C14	c20										
504	504	504	x	504	504										

6º ciclo **7º ciclo**

Figura 3.19a – Opção de troca "simultânea" de três ventosas: 6º ciclo trocando-se 501->504; 502->504; 501->504; 7º não há troca.

c5	c8	x	x	C13	c14	x	x	c20						
504	504	x	x	504	504	x	x	504						

6º ciclo **7º ciclo** **8º ciclo**

Figura 3.19b – Opção de troca "simultânea" de duas ventosas: 6º ciclo trocando-se 501->504; 502->504; 7º e 8º ciclos, não há trocas.

c5	x	x	x	c8	x	x	x	c13	x	x	x	c14	x	x	x	c20	x	x	x
504	x	x	x	504	x	x	x	504	x	x	x	504	x	x	x	504	x	x	x

6º ciclo **7º ciclo** **8º ciclo** **9ºciclo** **10º ciclo**

Figura 3.19c – Opção de troca de uma ventosa: 6º ciclo trocando-se 501->504; 7º, 8º, 9º e 10º ciclos, não há trocas.

As ilustrações da figura **3.19** conduzem a um aumento progressivo na quantidade de ciclos em função da progressiva menor quantidade disponível de *ventosas* para a captura e montagem dos componentes finais. Também foi dito no item **3.1.1**, que a forma de distribuição de *ventosas* sobre as cavidades do ATC tem influencia direta sobre qual quantidade de *ventosas* se deveriam trocar na mesma visita a este dispositivo. Portanto, a priorização do número de trocas relaciona-se dinamicamente com a seqüência avaliada, pois em um ciclo poderia ser melhor trocar as quatro ventosas, mas em algum ciclo posterior, poder-se-ia chegar à situação onde a troca de apenas uma *ventosa* seria mais adequada. Por exemplo, na situação hipotética ilustrada no item **3.1.2**, a troca das quatro *ventosas* só valeria a pena se fosse realmente simultânea, ou seja, se ao visitar o ATC, houvesse quatro cavidades vazias adjacentes para a devolução simultânea das *ventosas* 501, 502, 501 e 503. Além disso, seria igualmente necessário que existissem quatro *ventosas* do tipo 504 adjacentemente posicionados. Com tal coincidência, poder-se-ia afirmar que o tempo gasto para trocar as quatro *ventosas* equivaleria à troca de apenas uma... Contudo, em algum ciclo posterior em que fosse necessária a presença de quatro *ventosas* diferentes, poder-se-ia ter uma situação oposta, onde a existência da distribuição de *ventosas* do mesmo tipo, uma a o lado da outra, seria indesejável. Neste caso, seria bom se houvesse *ventosas* de tipos diferentes e adjacentemente distribuídas.

Como se trata de uma distribuição de *ventosas* fixa sobre o ATC, obter a melhor distribuição ou a melhor forma de priorizar a quantidade de trocas em uma mesma visita ao ATC, vai depender da seqüência particular avaliada, ou seja, depende de cada caso. Seria preciso decidir, ciclo após ciclo, qual tipo de prioridade usar. Para a seqüência analisada, e com a distribuição ilustrada da figura 3.5 do item **3.1.1**, chegou-se a conclusão que seria melhor trocar apenas uma *ventosa*, pois o algoritmo foi construído para deixar as trocas, se houver, para o final da seqüência, e, em geral, tratam-se de componentes com o mesmo tipo de *ventosa*.

Os principais resultados desse trabalho baseiam-se na seqüência de entrada de componentes mostrada na tabela 3.2. No capítulo de resultados serão mostrados, para essa seqüência de montagem, os tempos de montagem otimizados obtidos pelas heurísticas propostas, o tempo de montagem otimizado obtido pela máquina e os tempos de montagem obtidos por outros métodos anteriormente desenvolvidos.

Esta tabela é composta por nove colunas sendo que:

a) A coluna **Numeração** simplesmente enumera a listagem de componentes;

b) As colunas, **Coordenada x** e **Coordenada y,** apresentam os valores das coordenadas da PCI a ser montada, tomando-se como origem o canto inferior esquerdo da placa;

c) A coluna **giro** não foi usada neste trabalho, mas poderá ser utilizada em trabalhos futuros visando-se o aperfeiçoamento do modelo da máquina;

d) A coluna **Código do comp**. apresenta o código de identificação do tipo do componentes a ser montado;

e) A coluna **Cabeça** identifica qual a cabeça de inserção que capturará e montará o componente identificado por seu código;

f) A coluna ***Nozzle*** identifica qual *ventosa* deve ser utilizada pela cabeça para capturar e montar o componente;

g) A coluna **Seqüência** identifica qual a ordem seqüencial de montagem do componente identificado por seu código;

h) A coluna ***Slot*** identifica o *alimentador* ou *escaninho* que contém o componente a ser capturado e montado.

Tabela 3.2 – lista da seqüência de entrada de componentes

Numeração	Coordenada X	Coordenada y	giro	Código do componente	Cabeça	*Nozzle*	Seqüência de montagem	*Slot*
1	15.00	2.00	0.00	1-162-923-91	H1	501	2	36
2	24.00	2.00	0.00	1-216-067-91	H2	501	30	38
3	10.00	3.00	0.00	1-216-833-91	H3	502	43	40
4	24.00	6.00	0.00	1-218-289-91	H4	502	39	42
5	10.00	4.00	0.00	1-162-923-91	H1	501	09	36
6	19.00	2.00	0.00	1-216-067-91	H2	501	20	38
7	23.00	6.00	0.00	1-218-289-91	H3	502	37	42
8	20.00	6.00	0.00	1-218-289-91	H4	502	6	42
9	10.00	5.00	0.00	1-162-923-91	H1	501	10	36
10	11.00	6.00	0.00	1-216-067-91	H2	501	16	38
11	16.00	2.00	0.00	1-218-289-91	H3	502	5	42
12	13.00	6.00	0.00	1-218-289-91	H4	502	35	42
13	17.00	6.00	0.00	1-162-923-91	H1	502	1	36
14	18.00	3.00	0.00	1-216-067-91	H2	501	25	38
15	9.00	6.00	0.00	1-218-289-91	H3	502	33	42
16	9.00	5.00	0.00	1-218-289-91	H4	502	11	42
17	18.00	2.00	0.00	1-216-809-91	H4	503	23	26
18	8.00	4.00	0.00	1-216-186-91	H3	503	32	24
19	10.00	6.00	0.00	8-719-058-24	H2	504	12	22
20	16.00	4.00	0.00	1-242-772-91	H1	504	3	20
21	13.00	2.00	0.00	1-216-809-91	H4	503	18	26
22	11.00	4.00	0.00	1-216-186-91	H3	503	36	24

23	**16.00**	**5.00**	**0.00**	**8-719-058-24**	**H2**	**504**	**4**	**22**
24	**20.00**	**4.00**	**0.00**	**1-242-772-91**	**H1**	**504**	**8**	**20**
25	**12.00**	**5.00**	**0.00**	**1-216-809-91**	**H4**	**503**	**28**	**26**
26	**12.00**	**3.00**	**0.00**	**1-216-186-91**	**H3**	**503**	**34**	**24**
27	**22.00**	**5.00**	**0.00**	**8-719-058-24**	**H2**	**504**	**7**	**22**
28	**24.00**	**5.00**	**0.00**	**1-242-772-91**	**H1**	**504**	**13**	**20**
29	**8.00**	**3.00**	**0.00**	**8-729-920-75**	**H1**	**504**	**15**	**16**
30	**24.00**	**4.00**	**0.00**	**8-729-920-75**	**H2**	**504**	**29**	**18**
31	**23.00**	**5.00**	**0.00**	**1-216-186-91**	**H3**	**503**	**40**	**24**
32	**13.00**	**6.00**	**0.00**	**1-216-186-91**	**H4**	**503**	**38**	**24**
33	**18.00**	**5.00**	**0.00**	**8-729-920-75**	**H1**	**504**	**19**	**16**
34	**18.00**	**6.00**	**0.00**	**8-729-920-75**	**H2**	**504**	**24**	**18**
35	**8.00**	**5.00**	**0.00**	**1-216-809-91**	**H4**	**503**	**14**	**26**
36	**8.00**	**6.00**	**0.00**	**1-125-891-91**	**H2**	**504**	**31**	**8**
37	**12.00**	**6.00**	**0.00**	**1-125-891-91**	**H1**	**504**	**26**	**10**
38	**12.00**	**4.00**	**0.00**	**1-125-891-91**	**H2**	**504**	**44**	**8**
39	**24.00**	**3.00**	**0.00**	**1-125-891-91**	**H1**	**504**	**21**	**10**
40	**22.00**	**3.00**	**0.00**	**8-729-920-86**	**H1**	**504**	**22**	**48**
41	**22.00**	**4.00**	**0.00**	**8-729-920-86**	**H2**	**504**	**27**	**50**
42	**13.00**	**3.00**	**0.00**	**8-719-069-60**	**H2**	**504**	**17**	**6**
43	**22.00**	**6.00**	**0.00**	**8-729-424-02**	**H1**	**504**	**41**	**4**
44	**20.00**	**5.00**	**0.00**	**1-469-152-21**	**H1**	**504**	**42**	**2**

3.5 Considerações Finais

Os métodos apresentados no início deste capítulo e ilustrados através dos fluxogramas do item 3.1, representam a concepção original deste trabalho no sentido de solucionar o problema de otimização de montagem de componentes por meio de algoritmos simples e que mantêm o conceito de busca local inserido no método de subida de encosta com reinício aleatório. Programaram-se os algoritmos propostos em linguagem C++, por se tratar de uma linguagem estruturada, de rápido processamento e bastante robusta, dentre outras qualidades. Realizaram-se comparações entre os resultados obtidos por meio dos simuladores criados para os dois métodos e, também, entre os resultados obtidos por meio da proposta que pareceu ser mais satisfatória e os resultados obtidos com o otimizador comercial do próprio modelo da máquina usada no experimento prático. Este otimizador é conhecido pela sigla HLC, o qual, realiza otimizações se seqüências de montagem para o modelo de máquina JUKI-2030. Trata-se de um algoritmo bastante rápido e tem-se demonstrado eficiente. A vantagem do uso deste otimizador parece residir na quantidade de detalhes de informação dominados pelo fabricante, as quais não são facilmente obtidas. Contudo, um dos objetivos deste trabalho é realizar otimizações que sejam tão eficientes quanto às realizadas pelo próprio HLC (vide capítulo introdução – objetivos secundários).

Após programar-se os dois algoritmos e resolver-se a questão relacionada ao sistema de coordenadas adotado, aproveitou-se uma seqüência de montagem de componentes fictícia encontrada em [Craveiro 2005], e transcrita na tabela 3.2. O fato desta seqüência ser fictícia não traz prejuízo ao experimento, uma vez que a máquina JUKI pode operar em modo conhecido como *dry run*. Neste modo, todos os eventos são realizados pela máquina, como se fosse uma montagem real, porém, os componentes são fictícios, ou seja, as operações de captura e montagem, são realizadas no espaço vazio que deveria ser ocupado pelos componentes. Tal manobra deveu-se a pouca disponibilidade das máquinas para realização de experiências. Caso se optasse por uma operação real, o tempo para o carregamento de componentes nos *alimentadors*, e destes sobre os *escaninhos*, seria demasiado e insustentável para uma máquina de uso comercial, a qual foi cedida por curto espaço de tempo para validação dos resultados alcançados nos simuladores.

Cabe ressaltar também, que os simuladores implementados calcularam o tempo de montagem de componentes levando em conta os eventos abordados em 3.2.1. Um trabalho futuro, contudo, poderia abranger uma estratégia para contabilizar o tempo de deslocamento do bloco de cabeças quando este partiu da origem (ou repouso entre a montagem de uma PCI e a próxima) até o dispositivo de ATC para captura das *ventosas*. Este tempo, no entanto, foi considerado constante para qualquer seqüência de montagem. Ou seja, qualquer seqüência terá que partir da origem e se dirigir até o ATC para capturar as *ventosas*. Então, por se tratar de um intervalo de tempo inevitável, o mesmo não entrou no cômputo do tempo de montagem, tanto para os resultados dos algoritmos apresentados nos métodos 1 e 2, quanto para os resultados obtidos com o HLC. Portanto, a forma de cronometragem de tempo de montagem da seqüência testada (vide tabela3.2) foi rigorosamente a mesma para todos os otimizadores.

Além disso, os simuladores não se restringiram as regras dos tipos que foram encontras em alguns trabalhos pesquisados, como por exemplo, não é preciso haver apenas um *alimentador* por tipo de componente, ou seja, pode-se repetir um determinado tipo de componente em mais de um *alimentador*. A estratégia pretende comparar os resultados obtidos com e sem esta restrição, pelo menos no nível de simulações, conforme abordado no capítulo de resultados.

4. Resultados obtidos

Neste capítulo, a maioria dos resultados simulados, bem como dos resultados práticos, serão exibidos respeitando-se a ordem cronológica de obtenção dos mesmos. O objetivo principal para esta ordenação é mostrar, de forma clara, a lógica envolvida nas alterações que levaram aos resultados finais, pois, conforme mencionado no capítulo anterior, os resultados inicias não conduziram a resultados integralmente satisfatórios. Todavia, a despeito da substancial melhoria conquistada e apresentada na etapa final, não foram esgotadas todas as experiências vislumbradas, principalmente por falta de condições de aplicação prática continuada. Tais estratégias ficarão para abordagens futuras.

4.1 Interface do sistema

Nesta fase do trabalho não se enfatizou de forma alguma a apresentação estética da interface do sistema. Trata-se, portanto, de uma simples interface funcional, sobre a qual são realizadas poucas ações relativas às propostas 1 e 2. Conforme pode ser visualizado na figura 4.1, ainda há uma referência a uma terceira proposta (proposta 3), a qual já se mencionou no capítulo anterior. Esta proposta, diferencia-se da proposta 2, apenas por não permitir formação de um ciclo com ventosas repetidas. Seus resultados também serão mostrados, mas não houve vantagem evidente com relação à proposta2.

A figura 4.1 é composta por três botões de disparo dos algoritmos relativos às propostas 1, 2 e 3 acima informadas. Além destes há um quarto botão com a denominação “cancel” (nome padrão inerente ao MSVC++ 6.0), o qual tem a função de abortar ou interromper a proposta que tenha sido previamente disparada. A esquerda destes botões posicionam-se duas caixas de texto com os títulos “tempo mínimo” e “tempo máximo”, respectivamente. Tais objetos são atualizados com os respectivos tempos relativos à pior e a melhor otimização obtida durante o processamento das propostas. Ou seja, no final da simulação de uma proposta, o menor tempo de montagem (da melhor seqüência obtida) e o maior tempo de montagem (da pior

seqüência obtida) são exibidos. Juntamente com tais informações, dois gráficos também são atualizados. O gráfico superior, por questões de espaço, exibe no eixo horizontal apenas as iterações onde houve redução do tempo de montagem, e no eixo vertical o valor deste tempo em segundos. Desta forma, entre duas iterações consecutivas do gráfico, só há valores de tempo de montagem menores ou iguais ao antecessor. Por outro lado, o gráfico inferior, exibe todas as iterações do laço externo (o laço que define a quantidade de reinícios aleatórios - *estados*), porém, apenas o melhor vizinho (laço interno) de cada *estado*, também devido às restrições de espaço para exibição inteligível, é exibido. Pode-se observar neste gráfico o caráter aleatório do processamento, através dos "ziguezagues" presentes no gráfico. No eixo horizontal encontra-se a ordem da iteração até se atingir o critério de parada (quantidade de reinícios ou formação aleatória de *estados*) e no eixo vertical, encontra-se o valor relativo ao tempo de montagem do melhor vizinho do *estado* inicial em segundos.

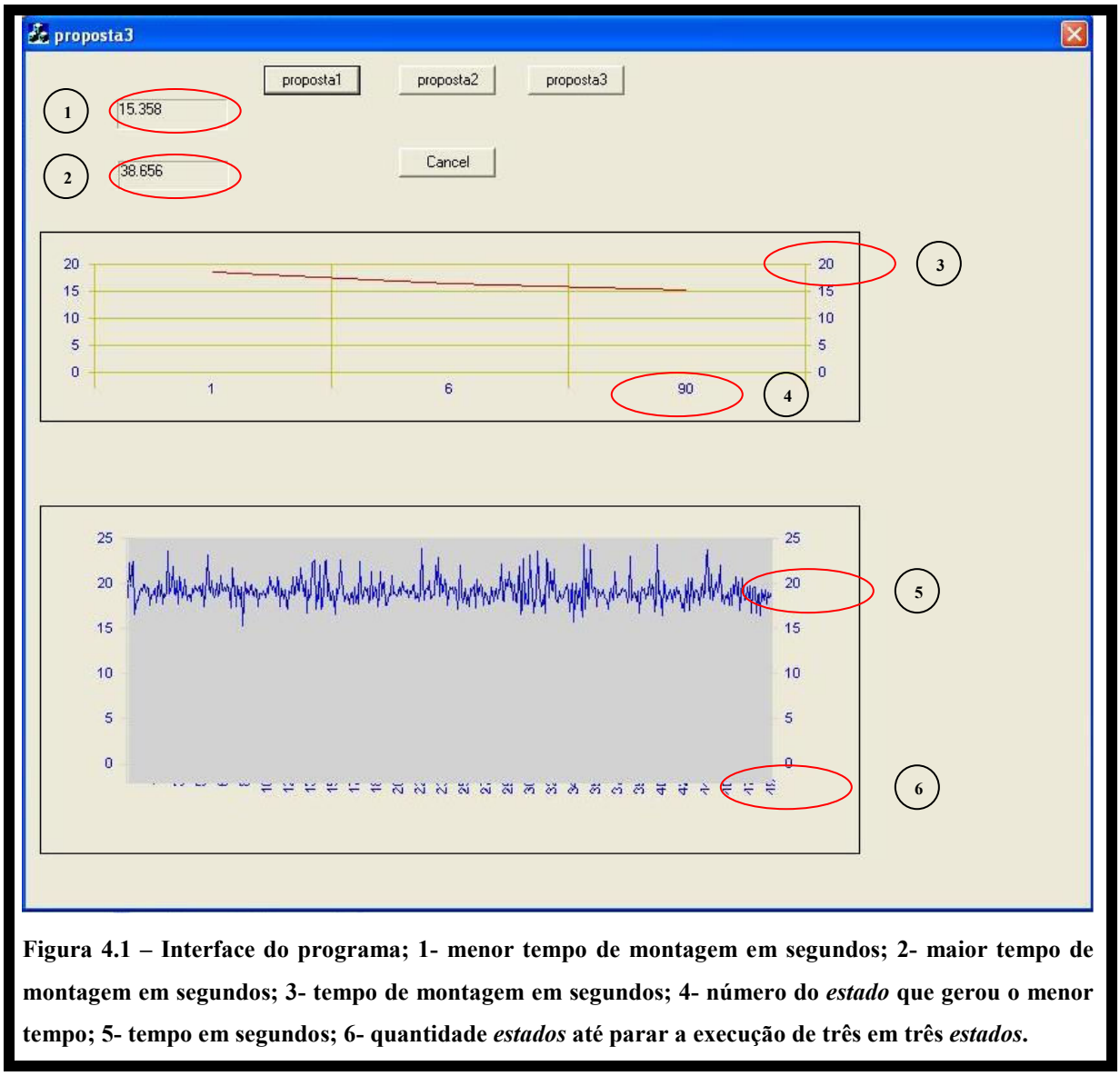

Figura 4.1 – Interface do programa; 1- menor tempo de montagem em segundos; 2- maior tempo de montagem em segundos; 3- tempo de montagem em segundos; 4- número do *estado* que gerou o menor tempo; 5- tempo em segundos; 6- quantidade *estados* até parar a execução de três em três *estados*.

Afora estes objetos, há a necessidade de acrescentarem-se outros objetos que permitam alterar-se a quantidade de vizinhos e de *estados aleatórios iniciais*, Lim_1 e Lim_2, respectivamente. Atualmente, tais constantes só podem ser definidas por meio de um arquivo de iniciação ou arquivo de cabeçalho (figura 4.2). Dentro deste arquivo, também se pode configurar a caminho (*path*) e nome do arquivo que contém a lista de componentes a montar, além de outras informações, conforme ilustradas na tabela 3.1. Outro valor configurável no arquivo de cabeçalho refere-se à quantidade de permutações na formação de um novo *estado*, por meio de uma constante que define a quantidade de permutações entre os componentes e outra constante que define a quantidade de permutações relativas aos *alimentadores*.

Figura 4.2 com a reprodução do arquivo de cabeçalho utilizado.

```
// proposta1Dlg.cpp : implementation file
//

#include "stdafx.h"
#include "proposta3.h"
#include "proposta3Dlg.h"

#include "component.h"
#include "math.h"
#include "stdio.h"
#include "conio.h"
#include "string.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

//inicio da seção de definições e variáveis globais
#define numSLOTS                60//30//60//20//120
#define numSEQUENCE             44//41//33//48//68
#define DUMMY                   8*numSEQUENCE
#define numINTERCHANGE          1
#define numNOZZLES              4//1//6
//#define typeCOMPONENTES       156//10//19//20
#define numCABECOTES            4
//#define numCiclos                     numSEQUENCE/numCABECOTES  //default, mas pode haver ciclos extras
devido as otimizações...

//especificações da máquina
#define velDoBraco              400//580.0//400//230//833.78             //mm/s
#define gapSlots                17.0                                    //mm
#define gapCabecotes            17.0                                    //mm
#define nSlotsADJACENTES        17.0                                   //mm
#define tempoCaptura            0.03008                                 //s
#define tempoMontagem           0.03008  //0.03008                      //s
#define nComprimentoMagazine    510.0                                   //mm
#define nDistanciaEntreMagazines  741.44                                //mm
#define nLarguraPCI             191.00//600.0                           //mm  // corresponde a coordenada y
#define nComprimentoPCI         162.00//400.0                           //mm  // corresponde a coordenada x
#define nOrigemSlotsX           -795.95//-214.2//nComprimentoPCI/2    // centro da placa
#define nOrigemSlotsY           -131.29//-126.93//nLarguraPCI/2         // centro da placa

#define ModuloY1                        5.0   //nDistanciaEntreMagazines/2  // distancia entre o centro e os magazines
                                        //(magazine atrás negativo; magazine a frente positivo)
#define ModuloY2                    746.44//640.15//746.44
```

```
#define OrigSlotX                    777.66//100.24//151.24          // coordenada X do 1º Slot
#define nTempoDeTrocaDeNozzle        2//*tempoCaptura//?             //s
#define ANCX                         915.48//278.79//267.67 nOrigemSlotsX     //mm
#define ANCY                         529.81//647.0//529.14//648.50 //647.00   //mm


#define numCoordenadasX              (int)(nLarguraPCI/10)      //divisor aleatorio
#define numCoordenadasY              (int)(nComprimentoPCI/10)  //divisor aleatorio
#define numVIZINHOS                  3000//100
#define numESTADOS                   3000//500

#define nCICLOS_SEM_ANC              5  //?

//OBS: Se numSEQUENCE == numCOMPONENTES então todos
//     os componentes da sequencia são IGUAIS!

//int nRelacaoNoozle[numSEQUENCE];
//int nRelacaoComponent[numSEQUENCE];
int nTypeNoozles[numNOOZLES];
int nSaveH[numCABECOTES+1];
int nTypeComponentes;
int nDescontarTempoCaptura;
double tempo=0;
int numNoozles=0,numDummy=numSEQUENCE,Seq=0,SaveSeq=0;//,nSeqTotal=0;
int q,dummy=0,saveNozzle[numCABECOTES];
bool bProposta2,bProposta1,troca=false,reotimizar=false;

CComponent* pComponent[numSEQUENCE+DUMMY];
CComponent* pSaveComponent[numSEQUENCE+DUMMY];
CComponent* pSaveComponent2[numSEQUENCE];
CComponent* pSaveVizinhoComponent[numSEQUENCE+DUMMY];
CComponent* pSaveSequenceComponent[numSEQUENCE+DUMMY];


//int ATC[20]={501,501,501,501,502,502,502,502,503,503,503,503,504,504,504,504,500,500,500,500};
//int qNDif,qNRest;



//final da seção de definições e variáveis globais
```

Figura 4.2 – ilustração do cabeçalho para variação dos parâmetros usados nas simulações

Finalmente, e antes de comentar sobre os resultados obtidos, denota-se mais uma vez que o ponto de partida para o processamento dos algoritmos encontra-se representado pelo que se convencionou chamar por *arquivo de entrada de dados*. Trata-

se de um simples arquivo texto, organizado em dez colunas e tantas linhas quanto forem os componentes a serem montados em uma seqüência. A figura 4.3 reproduz um destes arquivos e como se pode observar nada mais é do que a tabela 3.2 do capítulo 3. A diferença entre as informações constantes no arquivo de entrada e a tabela 3.2 reside sobre a segunda coluna do arquivo, a qual foi preenchida com um duplo x (xx) para todas as linhas. Assim foi deixado para que se pudesse inserir um novo tipo de dado no lugar do duplo x, expandindo-se, se preciso, a quantidade de informações analisadas pelas rotinas. Além daquela, há também a coluna referente ao giro dos componentes, a qual não está sendo usada até o presente. Ademais, deve-se observar que o arquivo de entrada não possui uma linha inicial indicando o tipo de dado como a que existe na tabela 3.2 e no *arquivo de saída de dados*.

```
1  xx  15.00   2.00  0.00  1-162-923-91  h1  501  2    36
2  xx  24.00   2.00  0.00  1-216-067-91  h2  501  30   38
3  xx  10.00   3.00  0.00  1-216-833-91  h3  502  43   40
4  xx  24.00   6.00  0.00  1-218-289-91  h4  502  39   42
5  xx  10.00   4.00  0.00  1-162-923-91  h1  501  9    36
6  xx  19.00   2.00  0.00  1-216-067-91  h2  501  20   38
7  xx  23.00   6.00  0.00  1-218-289-91  h3  502  37   42
8  xx  20.00   6.00  0.00  1-218-289-91  h4  502  6    42
9  xx  10.00   5.00  0.00  1-162-923-91  h1  501  10   36
10 xx  11.00  6.00  0.00  1-216-067-91  h2  501  16   38
11 xx  16.00  2.00  0.00  1-218-289-91  h 3  502  5    42
12 xx  13.00  6.00  0.00  1-218-289-91  h4   502  35  42
13 xx  17.00  6.00  0.00  1-162-923-91  h1   501  1    36
14 xx  18.00  3.00  0.00  1-216-067-91  h2   501  25  38
15 xx  9.00   6.00  0.00  1-218-289-91  h3   502  33  42
16 xx  9.00   5. 00 0.00  1-218-289-91  h4   502  11  42
17 xx  18.00  2.0   0.00  1-216-809-91  h4   503  23  26
18 xx  8.00   4.00  0.00  1-216-186-91  h3   503  32  24
19 xx  10.00  6.00  0.00  8-719-058-24  h2   504  12  22
20 xx  16.00  4.00  0.00  1-242-772-91  h1   504  3    20
21 xx  13.00  2.00  0.00  1-216-809-91  h4   503  18  26
22 xx  11.00  4.00  0.00  1-216-186-91  h3   503  36  24
23 xx  16.00  5.00  0.00  8-719-058-24  h2   504  4    22
24 xx  20.00  4.00  0.00  1-242-772-91  h1   504  8    20
25 xx  12.00  5.00  0.00  1-216-809-91  h4   503  28  26
26 xx  12.00  3.00  0.00  1-216-186-91  h3   503  34  24
27 xx  22.00  5.00  0.00  8-719-058-24  h2   504  7    22
28 xx  24.00  5.00  0.00  1-242-772-91  h1   504  13  20
29 xx  8.00   3.00  0.00  8-729-920-75  h1   504  15  16 x
30 xx  24.00  4 .00 0.00  8-729-920-75  h2   504  29  18 x
```

```
31 xx 23.00 5.00  0.00  1-216-186-91 h3  503 40  24
32 xx 13.00 6.00  0.00  1-216-186-91 h4  503 38  24
33 xx 18.00 5.00  0.00  8-729-920-75 h1  504 19  16 x
34 xx 18.00 6.00  0.00  8-729-920-75 h2  504 24  18 x
35 xx 8.00  5.00  0.00  1-216-809-91 h4  503 14  26
36 xx 8.00  6.00  0.00  1-125-891-91 h2  504 31  8 x
37 xx 12.00 6.00  0.00  1-125-891-91 h1  504 26  10 x
38 xx 12.00 4.00  0.00  1-125-891-91 h2  504 44  8 x
39 xx 24.00 3.00  0.00  1-125-891-91 h1  504 21  10 x
40 xx 22.00 3.00  0.00  8-729-920-86 h1  504 22  48 x
41 xx 22.00 4.00  0.00  8-729-920-86 h2  504 27  50 x
42 xx 13.00 3.00  0.00  8-719-069-60 h2  504 17  6
43 xx 22.00 6.00  0.00  8-729-424-02 h1  504 41  4
44 xx 20.00 5.00  0.00  1-469-152-21 h1  504 42  2
```

Figura 4.3 – ilustração do arquivo de entrada de dados. As linhas que terminam com a letra "x" sinalizam componentes dispostos em mais de um *escaninho.*

Findado o processamento, um arquivo de saída com a denominação "seqüência.txt" será automaticamente gerado. O nome deste arquivo pode ser alterado ou mesmo definido no arquivo de cabeçalho. Este arquivo é composto por oito colunas conforme ilustrado na figura 4.4.

Seq	Compo name	X	Y	Feeder	Nozzle	Dummy	Troca
1	1-125-891-91	12.0	6.0	38	504	FALSE	FALSE
2	1-242-772-91	16.0	4.0	26	504	FALSE	FALSE
3	1-216-067-91	24.0	2.0	40	501	FALSE	FALSE
4	1-216-809-91	12.0	5.0	10	503	FALSE	FALSE
5	8-719-069-60	3.0	5.0	50	504	FALSE	FALSE
6	8-719-058-24	16.0	5.0	24	504	FALSE	FALSE
7	1-162-923-91	15.0	2.0	22	501	FALSE	FALSE
8	1-216-809-91	18.0	2.0	10	503	FALSE	FALSE
9	1-242-772-91	20.0	4.0	26	504	FALSE	FALSE
10	1-242-772-91	24.0	5.0	26	504	FALSE	FALSE
11	1-162-923-91	10.0	4.0	22	501	FALSE	FALSE
12	1-216-186-91	8.0	4.0	20	503	FALSE	FALSE
13	8-719-058-24	22.0	5.0	24	504	FALSE	FALSE
14	8-729-920-75	18.0	6.0	16	504	FALSE	FALSE
15	1-216-067-91	19.0	2.0	40	501	FALSE	FALSE
16	1-216-809-91	13.0	2.0	10	503	FALSE	FALSE

17	1-125-891-91	8.0	6.0	42	504	FALSE	FALSE
18	1-125-891-91	24.0	3.0	38	504	FALSE	FALSE
19	1-162-923-91	10.0	5.0	22	501	FALSE	FALSE
20	1-216-186-91	11.0	4.0	20	503	FALSE	FALSE
21	8-729-920-75	24.0	4.0	16	504	FALSE	FALSE
22	8-729-920-86	22.0	4.0	48	504	FALSE	FALSE
23	1-216-067-91	11.0	6.0	40	501	FALSE	FALSE
24	1-216-186-91	12.0	3.0	20	503	FALSE	FALSE
25	8-729-920-75	18.0	5.0	8	504	FALSE	FALSE
26	8-729-920-86	22.0	3.0	18	504	FALSE	FALSE
27	1-162-923-91	17.0	6.0	22	501	FALSE	FALSE
28	1-216-186-91	23.0	5.0	20	503	FALSE	FALSE
29	8-729-920-75	8.0	3.0	8	504	FALSE	FALSE
30	1-125-891-91	12.0	4.0	42	504	FALSE	FALSE
31	1-216-067-91	18.0	3.0	40	501	FALSE	FALSE
32	1-216-809-91	8.0	5.0	10	503	FALSE	FALSE
33	8-719-058-24	10.0	6.0	24	504	FALSE	FALSE
34	8-729-424-02	22.0	6.0	36	504	FALSE	FALSE
35	1-218-289-91	23.0	6.0	4	502	TRUE	FALSE
36	1-216-186-91	13.0	6.0	20	503	FALSE	FALSE
37	1-469-152-21	20.0	5.0	2	504	FALSE	FALSE
38	1-469-152-21	20.0	5.0	2	502	TRUE	FALSE
39	1-469-152-21	20.0	5.0	2	502	TRUE	FALSE
40	1-469-152-21	20.0	5.0	2	502	TRUE	FALSE
41	1-216-833-91	10.0	3.0	6	502	FALSE	TRUE
42	1-218-289-91	24.0	6.0	4	502	FALSE	TRUE
43	1-218-289-91	23.0	6.0	4	502	FALSE	TRUE
44	1-218-289-91	20.0	6.0	4	502	FALSE	TRUE
45	1-218-289-91	16.0	2.0	4	502	FALSE	FALSE
46	1-218-289-91	13.0	6.0	4	502	FALSE	FALSE
47	1-218-289-91	9.0	6.0	4	502	FALSE	FALSE
48	1-218-289-91	9.0	5.0	4	502	FALSE	FALSE

Figura 4.4 – ilustração do arquivo de saída representando uma solução para o problema de otimização de seqüência de montagem de componentes.

a) A primeira coluna indica a seqüência de inserção dos componentes, ou seja, a ordem em que eles serão montados sobre a PCI;

b) A segunda coluna contém os códigos que identificam um determinado tipo de componente;

c) A terceira e a quarta colunas representam as coordenadas da PCI onde os componentes serão montados;

d) A quinta coluna representa a posição do alimentador no escaninho em numerais pares;

e) A sexta coluna representa o tipo de ventosa usado;

f) A sétima indica se a linha com dados relativos a um componente existe de fato (FALSE), ou se é fictícia (TRUE) e apenas para manter a estrutura de um ciclo;

g) A oitava coluna indica a posição seqüência onde será realizada uma troca de ventosa (TRUE), ou não (FALSE).

Este arquivo, a princípio, contém a melhor seqüência de montagem otimizada por um determinado algoritmo. O arquivo de entrada, contudo, corresponde ao *estado inicial* (aleatoriamente formado com todos os componentes a serem montados).

4.2 Resultados das Propostas sem otimizações

Neste tópico serão exibidos os resultados oriundos das propostas originais. Decidiu-se colocá-los aqui por questões de registro simplesmente... Os termos Proposta 1 e Heurística 1 são equivalentes. Da mesma forma, os termos proposta 2 e Heurística 2 são equivalentes. Tais resultados seguiram o seguinte roteiro:

a) Serão mostrados resultados da simulação dos dois algoritmos relativos às propostas 1 e 2 para seqüências com **10; 20 e 30** componentes;

b) Cada simulação do item anterior será repetida 10 vezes, registrando seus gráficos (interfaces com tempos de montagem), cronometrando-se também o tempo de processamento de cada uma, além da média e desvio padrão obtidos;
   Em cada simulação trabalhou-se com **500** *estados* aleatórios e com **100** vizinhos;

c) Para cada simulação dos itens **a** e **b** será exibida uma tabela de resultados destacando-se os menores tempos obtidos;

d) Para um dos métodos (o método mais promissor), repetir-se-á o processo acima, mas desta vez, usando-se uma seqüência com **44** componentes e: **5.000; 10.000; 15.000 e 20.000** *estados* aleatórios com **100** vizinhos cada;

e) Procurar-se-á, através dos gráficos, evidenciar o caráter de aleatoriedade do algoritmo de subida em encosta com reinício aleatório; Serão mostrados resultados de processos de otimização de uma seqüência com 44 componentes obtidas com o otimizador HLC e com o método mais promissor;

f) Sempre que possível serão mostrados resultados referentes ao tempo de montagem e ao tempo de processamento;

4.2.1 Tabelas de tempos simulados para a Proposta1 (itens 'a', 'b' referidos anteriormente).

As tabelas 4.1 e 4.2 resumem os resultados das simulações para os itens 'a' e 'b' conforme o roteiro mencionado no início desta seção.

Tabela 4.1 – tempos de montagens para 10 simulações com 10, 20 e 30 componentes – heurística 1.

nº. de componentes	**Tempo de montagem em segundos** para as simulações realizadas										Tempo médio	Desvio padrão Amostral
	1ª	**2ª**	**3ª**	**4ª**	**5ª**	**6ª**	**7ª**	**8ª**	**9ª**	**10ª**		
10	3,59	3,59	3,59	3,59	3,59	3,59	3,60	3,61	3,59	3,60	**3,59**	**~ 0**
20	10,69	10,57	10,48	10,50	10,44	10,39	10,61	10,48	11,2	10,48	**10,58**	**~ 0**
30	15,35	15,59	15,84	16,07	15,71	15,39	16,01	16,49	16,08	15,97	**15,85**	**~ 0**

Tabela 4.2 – tempos de processamentos para 10 simulações com 10, 20 e 30 componentes – heurística 1.

nº. de componentes	**Tempo de processamento em segundos** para as simulações realizadas										Tempo médio	Desvio padrão Amostral
	1ª	**2ª**	**3ª**	**4ª**	**5ª**	**6ª**	**7ª**	**8ª**	**9ª**	**10ª**		
10	2,5	2,5	2,5	2,5	2,5	2,5	2,5	2,5	2,5	2,5	**2,5**	**0**
20	4,0	4,0	4,0	4,0	4,0	4,0	4,0	4,0	4,0	4,0	**4,0**	**0**
30	7,0	7,0	7,0	7,0	7,0	7,0	7,0	7,0	7,0	7,0	**7,0**	**0**

4.2.2 Tabelas de tempos simulados para a Proposta2 (itens 'a', 'b' referidos anteriormente)

As tabelas 4.3 e 4.4 resumem os resultados das simulações para os itens 'a' e 'b' conforme o roteiro mencionado no início desta seção.

Tabela 4.3 – tempos de montagens para 10 simulações com 10, 20 e 30 componentes - heurística 2.

nº. de componentes	**Tempo de montagem em segundos** para as simulações realizadas										Tempo médio	Desvio padrão amostral
	1ª	**2ª**	**3ª**	**4ª**	**5ª**	**6ª**	**7ª**	**8ª**	**9ª**	**10ª**		
10	3,59	3,59	3,59	3,59	3,59	3,60	3,6	3,59	3,59	3,59	**3,59**	**~ 0**
20	10,55	10,60	11,20	10,35	10,65	10,48	10,52	10,46	10,69	10,56	**10,6**	**~ 0**
30	15,93	16,21	15,94	15,79	15,59	15,94	15,73	15,79	15,61	15,81	**15,83**	**~ 0**

Tabela 4.4 - tempos de processamentos para 10 simulações com 10, 20 e 30 componentes – heurística 2.

nº. de componentes	**Tempo de processamento em segundos** para as simulações realizadas										Tempo médio	Desvio padrão amostral
	1ª	**2ª**	**3ª**	**4ª**	**5ª**	**6ª**	**7ª**	**8ª**	**9ª**	**10ª**		
10	2,5	2,5	2,5	2,5	2,5	2,5	2,5	2,5	2,5	2,5	**2,5**	**0**
20	4,0	4,0	4,0	4,0	4,0	4,0	4,0	4,0	4,0	4,0	**4,0**	**0**
30	7,0	7,0	7,0	7,0	7,0	7,0	7,0	7,0	7,0	7,0	**7,0**	**0**

O item 'c' ficou prejudicado em função da grande similaridade dos tempos em todas as 10 simulações como comprovam as médias e os desvios padrão calculados.

4.2.3 Resultados relativos à Proposta 2 (item 'd' referido anteriormente)

Como os resultados exibidos nos itens 4.2.1 e 4.2.2 não apresentaram vantagens relativas que possibilitasse a escolha de uma proposta específica, elegeu-se a proposta 2 para realizar simulações relativas ao item 'c'. Os resultados exibidos a seguir são relativos a seqüência de montagem mostrada na figura 4.3, com **44 componentes**.

A tabela 4.5 mostra que dentre as simulações na faixa de 5.000 a 20.000 iterações, obteve-se o tempo de 23,23 segundos como menor tempo de montagem de 44 componentes ocorrido na sétima simulação com 10.000 iterações (10.000 reinícios de estados aleatórios) contra o maior tempo de montagem ocorrido na oitava simulação com 5.000 iterações.

Tabela 4.5 - tempos de montagens para 10 simulações com 44 componentes - heurística 2.

nº. de estados	**Tempo de montagem em segundos** para as simulações realizadas										Tempo médio	Desvio padrão
	1ª	**2ª**	**3ª**	**4ª**	**5ª**	**6ª**	**7ª**	**8ª**	**9ª**	**10ª**		
5.000	24,66s	24,85s	23,49s	24,51s	24,75s	24,45s	24,21s	**25,00s**	24,49s	24,96s	**24,53**	**0,442**
10.000	24,71s	24,10s	25,10s	24,57s	23,30s	24,68s	**23,23s**	24,34s	23,97s	23,96s	**24,19**	**0,607**
15.000	24,79s	24,49s	24,48s	24,16s	24,39s	24,36s	24,16s	23,36s	23,67s	23,55s	**24,14**	**0,508**
20.000	23,61s	24,10s	23,62s	24,10s	23,66s	23,36s	24,20s	23,31s	24,44s	23,91s	**23,83**	**0,376**

A tabela 4.6 mostra os tempos de processamento relativos a tabela 4.5 apresentada anteriormente.

Nesta tabela, observa-se que o tempo de processamento aumenta conforme o aumento do espaço avaliado passando de uma média de 1:39 minutos para 5.000 estados para 10:37 minutos para 20.000 estados. Porém, não há uma proporção linear.

Tabela 4.6 – tempos de processamentos para 10 simulações com 10, 20 e 30 componentes – heurística 2.

Nº. de estados	**Tempo de processamento em minutos:segundos** para as simulações realizadas										Tempo médio	Desvio padrão
	1ª	**2ª**	**3ª**	**4ª**	**5ª**	**6ª**	**7ª**	**8ª**	**9ª**	**10ª**		
5.000	1:38m	1:38m	1:40m	1:40m	1:40m	1:40m	1:40m	1:40m	1:40m	1:40m	**1:39**	-
10.000	3:49m	3:20m	3:40m	3:30m	3:30m	4:00m	3:30m	3:35m	4:00m	3:40m	**3:39**	-
15.000	5:38m	6:15m	6:48m	6:15m	5:16m	5:40m	5:39m	5:48m	6:10m	6:15m	**5:58**	-
20.000	8:46m	9:50m	9:41m	10:06m	10:16m	10:52m	11:10m	11:05m	11:55m	12:00m	**10:37**	-

Os próximos itens exibem as seqüências de entrada (com 10; 20; 30 e 44 componentes) e a seqüência de saída (resultado) relativa ao tempo de montagem de 23,23 segundos obtido. Esta simulação foi carregada em máquina real para a devida

comprovação prática além de comparação com resultados obtidos com uso do otimizador HLC para a mesma seqüência de entrada.

4.2.4 Alguns gráficos das simulações (item 'e')

As figuras 4.5 e 4.6 destacam em seus respectivos setores inferiores (em azul) o caráter de aleatoriedade durante o processo de busca da segunda proposta avaliada, com a montagem de 10 e 44 componentes, respectivamente.

Na figura 4.5 estão os resultados para uma busca por melhor seqüência de montagem de apenas 10 componentes. Neste caso, houve o reinício do processo (gerando um *estado* inicial aleatório) por 500 vezes. Para cada reinício, houve uma busca para até 100 vizinhos. Portanto, o total de iterações chegou a **50.000** com um tempo de processamento extremamente rápido como registrado na tabela 4.2 acima. Conforme exibido na figura, o menor tempo para a montagem dos 10 componentes seria **3,59** segundos contra um maior tempo de **11,51** segundos. A figura mostra ainda (no gráfico superior) dentre as 50.000 iterações, os pontos onde ocorreram melhorias nos resultados, ou seja, onde houve reduções do tempo de montagem. É significativo informar que o melhor resultado foi obtido muito cedo, como se pode observar no gráfico representado pela **175ª** iteração do total de 50.000. Isto quer dizer que o objetivo foi encontrado dentro dos primeiros **1,75%** das iterações.

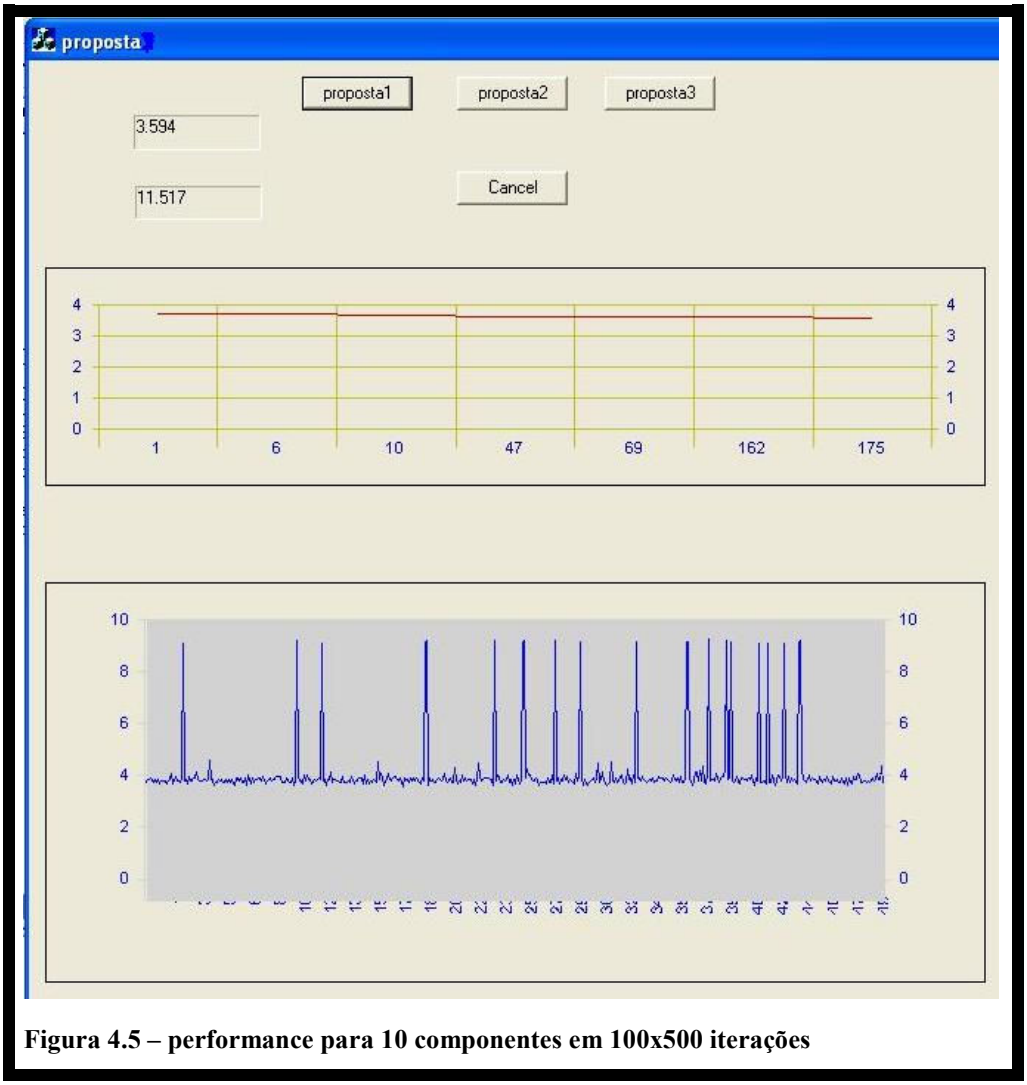

Figura 4.5 – performance para 10 componentes em 100x500 iterações

Já na figura 4.6, o número de iterações subiu para **1.000.000** (um milhão de iterações), resultantes de uma busca com 10.000 reinícios, cada qual com avaliação estendendo-se por até 100 vizinhos na busca pela melhor seqüência de montagem desta vez para 44 componentes. Como se pode observar, o gráfico inferior da figura apresenta uma compactação bem superior a da figura 4.5 na tentativa de exibir essas **10.000** iterações. O gráfico superior, entretanto, mostra que o melhor resultado (com tempo de **23,3** segundos) foi obtido na iteração **4.568**, mantendo desta forma, a mesma característica apresentada na figura 4.5, ou seja, o melhor resultado foi obtido muito cedo em relação ao um milhão de iterações que foram avaliadas. Isto representa apenas **0,45%** inicial do total de iterações. O tempo de processamento também se manteve reduzido, levando-se em conta a grande quantidade de iterações (vide tabela 4.6 acima).

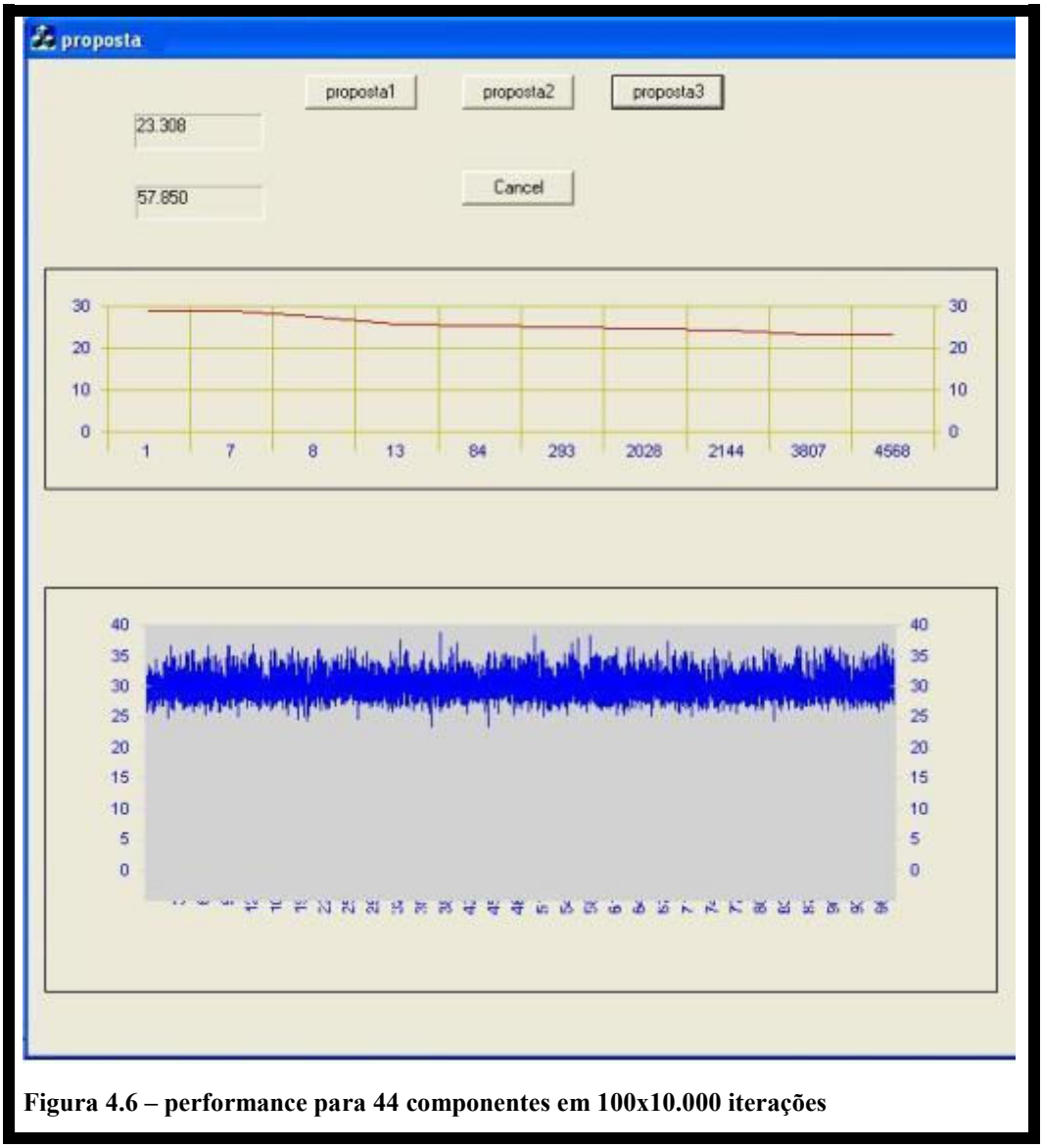

Figura 4.6 – performance para 44 componentes em 100x10.000 iterações

4.2.5 Comparação entre a Proposta 2 (escolhida) e o HLC (referente aos itens 'f', 'g' e 'h' conforme descritos no início desta seção)

Observe-se que a figura 4.7 é composta pelos mesmos elementos encontrados na figura 4.3 do capítulo anterior, porém com algumas alterações. A primeira diferença refere-se à seqüência de montagem, pois na figura 4.7 a seqüência de montagem (coluna "seq") é idêntica a ordem de leitura das linhas; Já na figura 4.3 isso não acontece. A segunda diferença refere-se à quantidade de ciclos e de linhas das listas. Ou seja, na figura 4.3 há apenas 44 linhas, ou seja, aquela seqüência corresponde a uma

montagem de 44 componentes; Contudo, na figura 4.7, percebe-se a existência de 53 linhas, mas a montagem continua reportando-se a 44 componentes (como não poderia ser diferente). As 9 linhas extras (com sinalização “sim” na última coluna) só existem para possibilitar a formação de ciclos com menos de 4 componentes e sua conformação, tanto com a estrutura do programa, quanto com o fato de que um ciclo só pode ser composto por cabeças diferentes. Portanto, não se pode ter h1-h1 no mesmo ciclo, por exemplo.

ordem fictício	--	X	Y	giro	componente	cabeça	ventosa	seq	escaninho
1	xx	15.00	2.00	0.00	1-162-923-91	h1	501	1	36
2	xx	24.00	2.00	0.00	1-216-067-91	h2	501	2	38
3	xx	10.00	3.00	0.00	1-216-833-91	h3	502	3	40
4	xx	24.00	6.00	0.00	1-218-289-91	h4	502	4	42
5	xx	10.00	4.00	0.00	1-162-923-91	h1	501	5	36
6	xx	19.00	2.00	0.00	1-216-067-91	h2	501	6	38
7	xx	23.00	6.00	0.00	1-218-289-91	h3	502	7	42
8	xx	20.00	6.00	0.00	1-218-289-91	h4	502	8	42
9	xx	10.00	5.00	0.00	1-162-923-91	h1	501	9	36
10	xx	11.00	6.00	0.00	1-216-067-91	h2	501	10	38
11	xx	16.00	2.00	0.00	1-218-289-91	h3	502	11	42
12	xx	13.00	6.00	0.00	1-218-289-91	h4	502	12	42
13	xx	17.00	6.00	0.00	1-162-923-91	h1	501	13	36
14	xx	18.00	3.00	0.00	1-216-067-91	h2	501	14	38
15	xx	9.00	6.00	0.00	1-218-289-91	h3	502	15	42
16	xx	9.00	5.00	0.00	1-218-289-91	h4	502	16	42
17	xx	18.00	2.0	0.00	1-216-809-91	h4	503	17	26
18	xx	8.00	4.00	0.00	1-216-186-91	h3	503	18	24
19	xx	10.00	6.00	0.00	8-719-058-24	h2	504	19	22
20	xx	16.00	4.00	0.00	1-242-772-91	h1	504	20	20
21	xx	13.00	2.00	0.00	1-216-809-91	h4	503	21	26
22	xx	11.00	4.00	0.00	1-216-186-91	h3	503	22	24
23	xx	16.00	5.00	0.00	8-719-058-24	h2	504	23	22
24	xx	20.00	4.00	0.00	1-242-772-91	h1	504	24	20
25	xx	12.00	5.00	0.00	1-216-809-91	h4	503	25	26
26	xx	12.00	3.00	0.00	1-216-186-91	h3	503	26	24

27	xx	22.00	5.00	0.00	8-719-058-24	h2	504	27	22	
28	xx	24.00	5.00	0.00	1-242-772-91	h1	504	28	20	
29	xx	8.00	3.00	0.00	8-729-920-75	h1	504	29	16	
30	xx	24.00	4.00	0.00	8-729-920-75	h2	504	30	18	
31	xx	23.00	5.00	0.00	1-216-186-91	h3	503	31	24	
32	xx	13.00	6.00	0.00	1-216-186-91	h4	503	32	24	
33	xx	18.00	5.00	0.00	8-729-920-75	h1	504	33	16	
34	xx	18.00	6.00	0.00	8-729-920-75	h2	504	34	18	
35	xx	8.00	5.00	0.00	1-216-809-91	h4	503	35	26	
36	xx	8.00	5.00	0.00	1-216-809-91	h4	503	36	26	sim
37	xx	8.00	6.00	0.00	1-125-891-91	h2	504	37	8	
38	xx	12.00	6.00	0.00	1-125-891-91	h1	504	38	10	
39	xx	12.00	6.00	0.00	1-125-891-91	h1	504	39	10	sim
40	xx	12.00	6.00	0.00	1-125-891-91	h1	504	40	10	sim
41	xx	12.00	4.00	0.00	1-125-891-91	h2	504	41	8	
42	xx	24.00	3.00	0.00	1-125-891-91	h1	504	42	10	
43	xx	24.00	3.00	0.00	1-125-891-91	h1	504	43	10	sim
44	xx	24.00	3.00	0.00	1-125-891-91	h1	504	44	10	sim
45	xx	22.00	3.00	0.00	8-729-920-86	h1	504	45	48	
46	xx	22.00	4.00	0.00	8-729-920-86	h2	504	46	50	
47	xx	22.00	4.00	0.00	8-729-920-86	h2	504	47	50	sim
48	xx	22.00	4.00	0.00	8-729-920-86	h2	504	48	50	sim
49	xx	13.00	3.00	0.00	8-719-069-60	h2	504	49	6	
50	xx	22.00	6.00	0.00	8-729-424-02	h1	504	50	4	
51	xx	22.00	6.00	0.00	8-729-424-02	h1	504	51	4	sim
52	xx	22.00	6.00	0.00	8-729-424-02	h1	504	52	4	sim
53	xx	20.00	5.00	0.00	1-469-152-21	h1	504	53	2	

Figura 4.7 – Seqüência de montagem inicial da proposta 1 (entrada)

A seguir estão expostas nas figuras 4.8 e 4.9, os resultados das otimizações realizadas pelo programa de implementação da heurística 2 e pelo programa comercial denominado HLC respectivamente. Estas soluções partiram (ou seja, tiveram como entrada inicial) a seqüência apresentada na figura 4.7.

Aproveita-se para destacar, no entanto, que a própria seqüência apresentada naquela figura já possui um caráter otimizado. Porém, a expectativa era que os otimizadores deveriam apresentar boas soluções independentes de estarem partindo de uma boa ou

má seqüência. Contudo, o tempo de processamento para alcançar tais soluções poderia variar em função da qualidade da entrada.

Conforme dito, a figura 4.8 representa a seqüência de montagem final e sua respectiva subdivisão em ciclos como solução encontrada pela heurística2.

Seq	componente	X	Y	Feeder	Nozzle	Fictício	Troca
1	1-125-891-91	12.0	6.0	38	504	FALSE	FALSE
2	1-242-772-91	16.0	4.0	26	504	FALSE	FALSE
3	1-216-067-91	24.0	2.0	40	501	FALSE	FALSE
4	1-216-809-91	12.0	5.0	10	503	FALSE	FALSE
5	8-719-069-60	13.0	3.0	50	504	FALSE	FALSE
6	8-719-058-24	16.0	5.0	24	504	FALSE	FALSE
7	1-162-923-91	15.0	2.0	22	501	FALSE	FALSE
8	1-216-809-91	18.0	2.0	10	503	FALSE	FALSE
9	1-242-772-91	20.0	4.0	26	504	FALSE	FALSE
10	1-242-772-91	24.0	5.0	26	504	FALSE	FALSE
11	1-162-923-91	10.0	4.0	22	501	FALSE	FALSE
12	1-216-186-91	8.0	4.0	20	503	FALSE	FALSE
13	8-719-058-24	22.0	5.0	24	504	FALSE	FALSE
14	8-729-920-75	18.0	6.0	16	504	FALSE	FALSE
15	1-216-067-91	19.0	2.0	40	501	FALSE	FALSE
16	1-216-809-91	13.0	2.0	10	503	FALSE	FALSE
17	1-125-891-91	8.0	6.0	42	504	FALSE	FALSE
18	1-125-891-91	24.0	3.0	38	504	FALSE	FALSE
19	1-162-923-91	10.0	5.0	22	501	FALSE	FALSE
20	1-216-186-91	11.0	4.0	20	503	FALSE	FALSE
21	8-729-920-75	24.0	4.0	16	504	FALSE	FALSE
22	8-729-920-86	22.0	4.0	48	504	FALSE	FALSE
23	1-216-067-91	11.0	6.0	40	501	FALSE	FALSE
24	1-216-186-91	12.0	3.0	20	503	FALSE	FALSE
25	8-729-920-75	18.0	5.0	8	504	FALSE	FALSE
26	8-729-920-86	22.0	3.0	18	504	FALSE	FALSE
27	1-162-923-91	17.0	6.0	22	501	FALSE	FALSE
28	1-216-186-91	23.0	5.0	20	503	FALSE	FALSE
29	8-729-920-75	8.0	3.0	8	504	FALSE	FALSE

30	1-125-891-91	12.0	4.0	42	504	FALSE	FALSE
31	1-216-067-91	18.0	3.0	40	501	FALSE	FALSE
32	1-216-809-91	8.0	5.0	10	503	FALSE	FALSE
33	8-719-058-24	10.0	6.0	24	504	FALSE	FALSE
34	8-729-424-02	22.0	6.0	36	504	FALSE	FALSE
35	1-218-289-91	23.0	6.0	4	502	TRUE	FALSE
36	1-216-186-91	13.0	6.0	20	503	FALSE	FALSE
37	1-469-152-21	20.0	5.0	2	504	FALSE	FALSE
38	1-218-289-91	9.0	6.0	2	502	TRUE	FALSE
39	1-218-289-91	24.0	6.0	2	502	TRUE	FALSE
40	1-218-289-91	20.0	6.0	2	502	TRUE	FALSE
41	1-216-833-91	10.0	3.0	6	502	FALSE	TRUE
42	1-218-289-91	24.0	6.0	4	502	FALSE	TRUE
43	1-218-289-91	23.0	6.0	4	502	FALSE	TRUE
44	1-218-289-91	20.0	6.0	4	502	FALSE	TRUE
45	1-218-289-91	16.0	2.0	4	502	FALSE	FALSE
46	1-218-289-91	13.0	6.0	4	502	FALSE	FALSE
47	1-218-289-91	9.0	6.0	4	502	FALSE	FALSE
48	1-218-289-91	9.0	5.0	4	502	FALSE	FALSE

Figura 4.8 – Seqüência de montagem final da heurística 2 (saída)

A figura 4.9 a seguir, representa a seqüência de montagem final e sua respectiva subdivisão em ciclos conforme a solução encontrada pelo algoritmo HLC.

ordem	--	X	Y	giro	componente	cabeça	ventosa	seq	escaninho	fictício
1	xx	8.00	4.00	0.00	1-216-186-91	h1	503	1	20	
2	xx	15.00	2.00	0.00	1-162-923-91	h2	501	2	22	
3	xx	22.00	5.00	0.00	8-719-058-24	h3	504	3	24	
4	xx	24.00	5.00	0.00	1-242-772-91	h4	504	4	26	
5	xx	12.00	3.00	0.00	1-216-186-91	h1	503	5	20	
6	xx	17.00	6.00	0.00	1-162-923-91	h2	501	6	22	
7	xx	16.00	5.00	0.00	8-719-058-24	h3	504	7	24	
8	xx	20.00	4.00	0.00	1-242-772-91	h4	504	8	26	
9	xx	11.00	4.00	0.00	1-216-186-91	h1	503	9	20	
10	xx	10.00	4.00	0.00	1-162-923-91	h2	501	10	22	
11	xx	10.00	6.00	0.00	8-719-058-24	h3	504	11	24	

12	xx	16.00	4.00	0.00	1-242-772-91	h4	504	12	26	
13	xx	18.00	2.00	0.00	1-216-809-91	h1	503	13	10	
14	xx	24.00	2.00	0.00	1-216-067-91	h2	501	14	40	
15	xx	24.00	3.00	0.00	1-125-891-91	h3	504	15	42	
16	xx	24.00	4.00	0.00	8-729-920-75	h4	504	16	16	
17	xx	13.00	2.00	0.00	1-216-809-91	h1	503	17	10	
18	xx	18.00	3.00	0.00	1-216-067-91	h2	501	18	40	
19	xx	12.00	4.00	0.00	1-125-891-91	h3	504	19	42	
20	xx	18.00	5.00	0.00	8-729-920-75	h4	504	20	16	
21	xx	12.00	5.00	0.00	1-216-809-91	h1	503	21	10	
22	xx	19.00	2.00	0.00	1-216-067-91	h2	501	22	40	
23	xx	12.00	6.00	0.00	1-125-891-91	h3	504	23	42	
24	xx	18.00	6.00	0.00	8-729-920-75	h4	504	24	16	
25	xx	8.00	5.00	0.00	1-216-809-91	h1	503	25	10	
26	xx	11.00	6.00	0.00	1-216-067-91	h2	501	26	40	
27	xx	8.00	6.00	0.00	1-125-891-91	h3	504	27	42	
28	xx	8.00	3.00	0.00	8-729-920-75	h4	504	28	16	
29	xx	23.00	5.00	0.00	1-216-186-91	h1	503	29	20	
30	xx	10.00	5.00	0.00	1-162-923-91	h2	501	30	22	
31	xx	22.00	4.00	0.00	8-729-920-86	h3	504	31	48	
32	xx	13.00	3.00	0.00	8-719-069-60	h4	504	32	50	
33	xx	13.00	6.00	0.00	1-216-186-91	h1	503	33	20	
ordem	--	X	Y	giro	componente	cabeça	ventosa	seq	escaninho	fictício
34	xx	16.00	2.00	0.00	1-218-289-91	h2	500	34	4	(troca)
35	xx	22.00	3.00	0.00	8-729-920-86	h3	504	35	48	
36	xx	22.00	6.00	0.00	8-729-424-02	h4	504	36	36	
37	xx	9.00	5.00	0.00	1-218-289-91	h1	502	37	4	sim
38	xx	9.00	5.00	0.00	1-218-289-91	h2	502	38	4	
39	xx	20.00	5.00	0.00	1-469-152-21	h3	504	39	2	
40	xx	9.00	5.00	0.00	1-218-289-91	h4	502	40	4	sim
41	xx	9.00	5.00	0.00	1-218-289-91	h1	502	41	4	sim
42	xx	9.00	6.00	0.00	1-218-289-91	h2	502	42	4	
43	xx	9.00	5.00	0.00	1-218-289-91	h3	502	43	4	sim
44	xx	9.00	5.00	0.00	1-218-289-91	h4	502	44	4	sim
45	xx	9.00	5.00	0.00	1-218-289-91	h1	502	45	4	sim
46	xx	13.00	6.00	0.00	1-218-289-91	h2	502	46	4	
47	xx	9.00	5.00	0.00	1-218-289-91	h3	502	47	4	sim
48	xx	9.00	5.00	0.00	1-218-289-91	h4	502	48	4	sim
49	xx	9.00	5.00	0.00	1-218-289-91	h1	502	49	4	sim

50	xx	20.00	6.00	0.00	1-218-289-91	h2	502	50	4	
51	xx	9.00	5.00	0.00	1-218-289-91	h3	502	51	4	sim
52	xx	9.00	5.00	0.00	1-218-289-91	h4	502	52	4	sim
53	xx	9.00	5.00	0.00	1-218-289-91	h1	502	53	4	sim
54	xx	23.00	6.00	0.00	1-218-289-91	h2	502	54	4	
55	xx	9.00	5.00	0.00	1-218-289-91	h3	502	55	4	sim
56	xx	9.00	5.00	0.00	1-218-289-91	h4	502	56	4	sim
57	xx	9.00	5.00	0.00	1-218-289-91	h1	502	57	4	sim
58	xx	24.00	6.00	0.00	1-218-289-91	h2	502	58	4	
59	xx	9.00	5.00	0.00	1-218-289-91	h3	502	59	4	sim
60	xx	9.00	5.00	0.00	1-218-289-91	h4	502	60	4	sim
61	xx	9.00	5.00	0.00	1-218-289-91	h1	502	61	4	sim
62	xx	10.00	3.00	0.00	1-216-833-91	h2	502	62	6	
63	xx	9.00	5.00	0.00	1-218-289-91	h3	502	63	4	sim
64	xx	9.00	5.00	0.00	1-218-289-91	h4	502	64	4	sim

Figura 4.9 – Seqüência de montagem final do HLC

Observe-se que a figura 4.9 é composta por 64 linhas onde se destacam transições alternadas entre as cores azul e preto de quatro em quatro linhas. Estas transições servem para separar os ciclos formados durante o processo de otimização do algoritmo HLC. Relembra-se que a seqüência de partida esta representada na figura 4.7.

A coluna “fictício” sinaliza uma linha ou componente que na realidade não existe, identificado pela inscrição “sim” sobre a mesma e servindo apenas para preencher a estrutura do programa criado, o qual representa os ciclos contendo quatro componentes.

Desta forma, a partir da linha 37, observam-se ciclos com menos de quatro componentes. Notoriamente, os seis últimos ciclos possuem somente um único componente a ser montado usando-se a cabeça *h2* com *ventosa* 502. Também se observe que estes ciclos montam o mesmo componente que estariam presentes no *escaninho* 4.

Finalmente, deve ser registrado que houve apenas uma troca de *ventosa* conforme indicado na linha 34.

A partir das informações acima, traçou-se um comparativo entre os resultados obtidos com a heurística 2 e com o otimizador comercial HLC. Tal comparação foi resumida na tabela 4.7.

Os critérios avaliados foram:

1- QCG – Quantidade de ciclos gerados;
2- QC4Cap – Quantidade de ciclos com quatro capturas simultâneas de componentes;
3- QC3Cap – Quantidade de ciclos com três capturas simultâneas de componentes;
4- QC2Cap – Quantidade de ciclos com duas capturas simultâneas de componentes;
5- QC1Cap – Quantidade de ciclos com uma captura de componente;
6- QTVS – Quantidade de trocas de ventosas simultâneas;
7- QVATC – Quantidade de visitas ao ATC;
8- TM – Tempo de montagem dos componentes;
9- TP – Tempo de processamento para encontrar as soluções;
10- TV – Tempo total para trocar todas as *ventosas*;

A tabela 4.7 mostra que o algoritmo HLC obteve uma solução bastante superior a que foi gerada pela heurística 2. Como exibido na coluna TM, o tempo de montagem dos mesmos 44 componentes foi de 22,91 segundos para o HLC contra 28,38 segundos para o algoritmo da heurística 2. Portanto, houve uma diferença substancial de 5,97 segundos a favor do HLC, ou seja, um tempo 19,2 % menor para montar a mesma quantidade de componentes...

Tabela 4.7 – comparação entre as soluções obtidas pela heurística 2 e HLC

	QCG	QC4Cap	QC3Cap	QC2Cap	QC1Cap	QTVS	QVATC	TM	TP	TV
HLC	16	3	0	10	6	1	1	22,91s	30s	1,98s
Heurística2	12	0	0	2	1	4	1	28,38s	10s	5,42s

A heurística 2 encontrou a solução mostrada na tabela 4.7 dentro de um universo de combinações aleatórias que envolveu 500 *estados* com 100 vizinhos cada. Tal escolha adveio da constatação, durante o processo de simulações, que não se conseguiria melhorar a solução de forma significativa simplesmente aumentando-se o espaço percorrido pelo algoritmo. Isso corresponderia a aumentar a quantidade de estados e/ou sua vizinhança. Tal situação pode ser comprovada pelos resultados exibidos na tabela 4.7 anterior.

Analisando os demais critérios abortados na tabela 4.7, observa-se:

Houve vantagem para a heurística 2 quanto ao:

1) QCG;
2) TP;
3) QTVS;

Houve desvantagem quanto ao:

1) QC4Cap;
2) QC2Cap;
3) QC1Cap;
4) TM;
5) TV;

Houve empate quanto ao:

1) QC3Cap;
2) QVATC;

É relativamente fácil observar que a quantidade itens em desvantagem foi bastante superior à quantidade de itens em vantagem obtidos com o algoritmo da heurística 2. A razão para tal desvantagem, e conseqüente ineficiência do método quando comparado ao HLC deveu-se ao menor número de capturas simultâneas (QCxCap) e o tempo total gasto na operação de troca de *ventosas* (TM).

Esta foi a razão pela qual se realizaram as otimizações descritas nos itens 3.3 e 3.4, as quais foram responsáveis pela melhoria substancial da eficiência do algoritmo (heurística 2). Antes de explorar os resultados obtidos com essas otimizações,

esclarece-se que em função das mesmas, não houve um tratamento especial que envolvesse o balanceamento de carga de trabalho entre as cabeças de inserção de componentes, uma vez que o objetivo final é reduzir o tempo de montagem final e, se possível, com pouco tempo de processamento na busca da solução. Além disso, é interessante observar que a redução na quantidade de ciclos a favor da heurística 2, com uma redução de 14 para 12 ciclos, não significou uma obrigatória redução do tempo de montagem, TM.

A figura 4.10 exibe como ficaram os gráficos após atingidos os critérios de parada da heurística 2 para a simulação analisada. Nesta figura, pode-se observar na caixa superior da esquerda o tempo de montagem alcançado com o valor de 28,315 segundos. Este menor tempo representando a melhor solução obtida contrasta com os 65,334 segundos obtidos para o pior caso, conforme a caixa logo abaixo da anterior.

O gráfico superior mostra que a convergência para melhor seqüência de montagem foi atingida com apenas seis pontos de mudanças sucessivamente melhores que o ponto de partida, o qual estava na casa dos 32 segundos. Este gráfico filtra todos os tempos maiores (piores) que foram obtidos durante o processamento.

Já o gráfico inferior mostra o caráter aleatório do processo, e portanto, exibe todo o espaço varrido durante a busca na vizinhança do estado inicial.

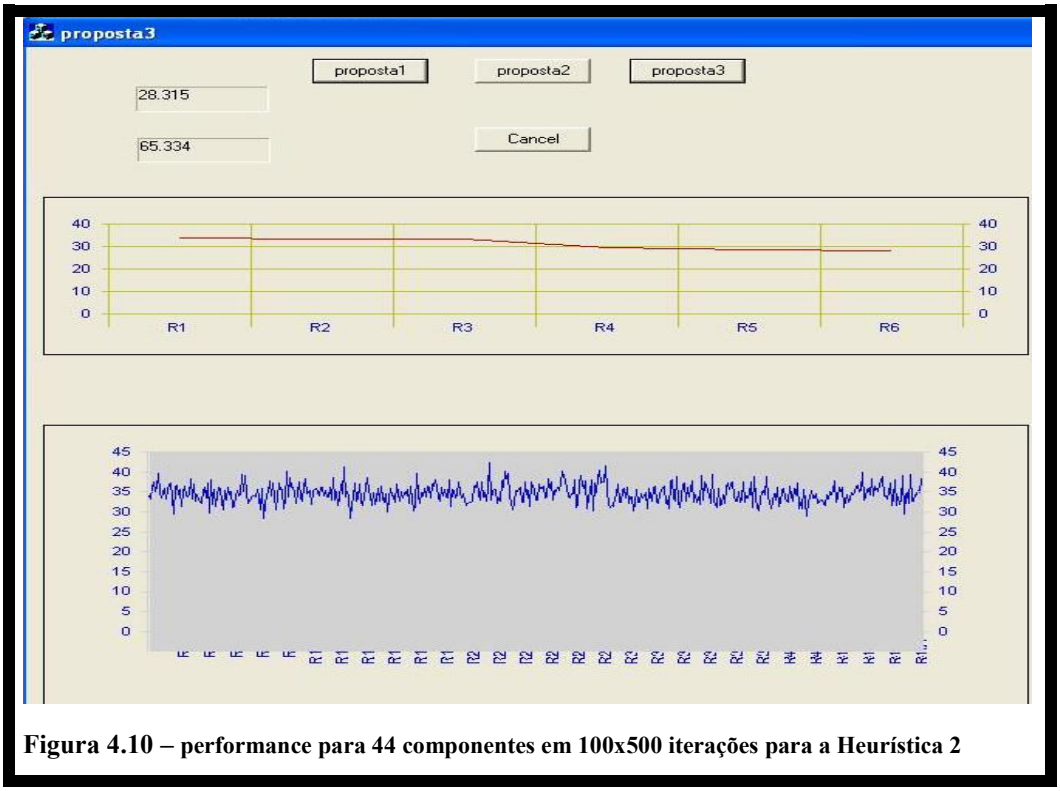

Figura 4.10 – performance para 44 componentes em 100x500 iterações para a Heurística 2

4.3 Resultados da Heurística 2 com otimizações

Nesta seção, mostrar-se-á o resultado alcançado depois que as otimizações mencionadas nas seções 3.3 e 3.4 foram acrescentadas a heurística 2. Porém, a despeito da comparação que será exibida com relação ao HLC, o enfoque também traçara uma comparação com resultados obtidos com trabalhos recentes nesta mesma área. A saber: o algoritmo genético elaborado por Craveiro, Carlos 2004 e o algoritmo memético elaborado por Carvalho, Elidelson 2007. Ambos os trabalhos desenvolvidos nos programas de mestrado vinculados a Universidade do Amazonas na área de engenharia elétrica. Trata-se de algoritmos capazes de resolver os mesmos tipos de problemas abordados pelo trabalho presente. O objetivo não é apontar qual dos trabalhos seria mais eficiente, mas simplesmente tomá-los como referencia na validação da heurística baseada no uso do algoritmo *hill climbing* no sentido de resolver o mesmo problema com objetivos citados durante a introdução.

4.3.1 Nova solução proposta pela Heurística 2

A listagem mostrada na figura 4.11 representa uma das soluções obtidas com a heurística 2. Foram realizadas diversas simulações variando-se os parâmetros do algoritmo no que tange a permissão ou não de *ventosas* repetidas em um mesmo ciclo de captura-montagem (*pick&placement*), a quantidade de reinícios aleatórios de *estados*, a quantidade de *estados* vizinhos daqueles *estados*, e até mesmo a quantidade de permutações envolvidas na formação de um novo *estado*. A listagem representa apenas um destes resultados e não foi necessariamente o melhor resultado de simulação obtido. Porém seus dados já demonstraram uma melhoria de performance quanto aos resultados obtidos na seção 4.2.5. Tal listagem representa o processamento de 2.500 *estados* com 5.000 vizinhos cada um, ou seja, um total de **12.500.000** combinações! Tal ordem de grandeza representa um exagero desnecessário como demonstraram os resultados obtidos com uma quantidade bem menor de combinações. Mas na prática, este foi o resultado escolhido para ser carregado na máquina de inserção automática para comprovação prática da simulação. Então, o tempo de processamento até encontrá-lo foi um pouco elevado em função do grande número de iterações envolvidas e ficou em torno dos 30 minutos sem qualquer tipo de otimização de lógica de programação relativa aos laços e operações de comparação envolvidos.

Além disso, cabe registrar que a solução ora comentada, foi gerada, usando-se como critérios:

1) A permissão de única troca de *ventosa* durante uma mesma visita ao ATC; Apesar de ter-se realizado outros testes com permissão de uma quantidade maior de trocas.

2) A permissão de *ventosas* repetidas (*ventosas* iguais) em um mesmo ciclo de captura-montagem; Apesar de ter-se realizado outros testes sem a permissão de *ventosas* iguais em um mesmo ciclo.

Cabe ainda esclarecer que as linhas cuja coluna “compo name” estão preenchidas com uma seqüência da letra x, são na verdade posições vazias dentro de um mesmo

ciclo como sinalizado pela coluna “dummy” com a inscrição TRUE. Este tipo de notação serviu apenas durante o processo de debug do algoritmo. Por exemplo, a linha 59 possui tal característica, e, como os ciclos são formados por quatro linhas de componentes consecutivas (nas quais as cabeças *h1*, *h2*, *h3* e *h4* estão associadas da menor para a maior linha dentro do ciclo), então apenas a linha 57 carregaria o componente transportado pela cabeça *h1;* Ou seja, as linhas 58, 59 e 60 indicam que as cabeças *h2*, *h3* e *h4* não carregariam componentes no ciclo 15 na listagem.

Seq	Compo name	X	Y	Feeder	Nozzle	Dummy	Troca
1	8-729-920-75	18.0	6.0	50	504	FALSE	FALSE
2	1-216-067-91	11.0	6.0	36	501	FALSE	FALSE
3	1-218-289-91	23.0	6.0	38	502	FALSE	FALSE
4	1-216-186-91	23.0	5.0	40	503	FALSE	FALSE
5	1-242-772-91	20.0	4.0	16	504	FALSE	FALSE
6	1-162-923-91	15.0	2.0	18	501	FALSE	FALSE
7	1-218-289-91	24.0	6.0	38	502	FALSE	FALSE
8	1-216-809-91	18.0	2.0	26	503	FALSE	FALSE
9	8-719-058-24	10.0	6.0	10	504	FALSE	FALSE
10	1-162-923-91	10.0	4.0	18	501	FALSE	FALSE
11	1-216-833-91	10.0	3.0	42	502	FALSE	FALSE
12	1-216-186-91	8.0	4.0	40	503	FALSE	FALSE
13	1-242-772-91	24.0	5.0	16	504	FALSE	FALSE
14	1-162-923-91	10.0	5.0	18	501	FALSE	FALSE
15	1-218-289-91	20.0	6.0	38	502	FALSE	FALSE
16	1-216-809-91	13.0	2.0	26	503	FALSE	FALSE
17	8-729-920-75	24.0	4.0	50	504	FALSE	FALSE
18	1-216-067-91	24.0	2.0	36	501	FALSE	FALSE
19	1-218-289-91	16.0	2.0	38	502	FALSE	FALSE
20	1-216-186-91	11.0	4.0	40	503	FALSE	FALSE
21	8-729-920-75	18.0	5.0	48	504	FALSE	FALSE
22	1-216-067-91	19.0	2.0	36	501	FALSE	FALSE
23	1-218-289-91	13.0	6.0	38	502	FALSE	FALSE
24	1-216-186-91	13.0	6.0	40	503	FALSE	FALSE
25	1-125-891-91	8.0	6.0	24	504	FALSE	FALSE
26	1-162-923-91	17.0	6.0	18	501	FALSE	FALSE
27	1-218-289-91	9.0	6.0	38	502	FALSE	FALSE
28	1-216-809-91	12.0	5.0	26	503	FALSE	FALSE
29	1-125-891-91	12.0	4.0	24	504	FALSE	FALSE
30	1-216-067-91	18.0	3.0	36	501	FALSE	FALSE
31	1-218-289-91	9.0	5.0	38	502	FALSE	FALSE
32	1-216-186-91	12.0	3.0	40	503	FALSE	FALSE
33	8-729-920-86	22.0	4.0	6	504	FALSE	FALSE

34	8-719-058-24	22.0	5.0	10	504	TRUE	FALSE
35	8-729-920-86	22.0	3.0	2	504	TRUE	FALSE
36	1-216-809-91	8.0	5.0	26	503	FALSE	FALSE
37	1-469-152-21	20.0	5.0	4	504	FALSE	FALSE
38	8-729-920-75	8.0	3.0	48	504	TRUE	FALSE
39	1-125-891-91	24.0	3.0	20	504	TRUE	FALSE
40	8-729-424-02	22.0	6.0	22	504	TRUE	FALSE
41	8-719-069-60	13.0	3.0	8	504	FALSE	FALSE
42	1-242-772-91	16.0	4.0	16	504	TRUE	FALSE
43	1-125-891-91	12.0	6.0	20	504	TRUE	FALSE
44	8-719-058-24	16.0	5.0	10	504	TRUE	FALSE
45	1-242-772-91	16.0	4.0	16	504	FALSE	FALSE
46	8-719-058-24	16.0	5.0	10	504	TRUE	FALSE
47	-719-058-24	22.0	5.0	10	504	TRUE	FALSE
48	8-729-920-75	8.0	3.0	48	504	TRUE	FALSE
49	1-125-891-91	12.0	6.0	20	504	FALSE	FALSE
50	1-125-891-91	24.0	3.0	20	504	TRUE	FALSE
51	8-729-920-86	22.0	3.0	2	504	TRUE	FALSE
52	8-729-424-02	22.0	6.0	22	504	TRUE	FALSE
53	8-719-058-24	16.0	5.0	10	504	FALSE	FALSE
54	8-719-058-24	22.0	5.0	10	504	TRUE	FALSE
55	8-729-920-75	8.0	3.0	48	504	TRUE	FALSE
56	1-125-891-91	24.0	3.0	20	504	TRUE	FALSE
57	8-729-920-86	22.0	3.0	2	504	FALSE	FALSE
58	8-729-424-02	22.0	6.0	22	504	TRUE	FALSE
59	XXXXXXX	0.0	0.0	0	0	TRUE	FALSE
60	XXXXXXX	0.0	0.0	0	0	TRUE	FALSE
61	8-719-058-24	22.0	5.0	10	504	FALSE	FALSE
62	8-729-920-75	8.0	3.0	48	504	TRUE	FALSE
63	1-125-891-91	24.0	3.0	20	504	TRUE	FALSE
64	8-729-424-02	22.0	6.0	22	504	TRUE	FALSE
65	8-729-920-75	8.0	3.0	48	504	FALSE	FALSE
66	1-125-891-91	24.0	3.0	20	504	TRUE	FALSE
67	8-729-424-02	22.0	6.0	22	504	TRUE	FALSE
68	XXXXXXXX	0.0	0.0	0	0	TRUE	FALSE
69	1-125-891-91	24.0	3.0	20	504	FALSE	FALSE
70	8-729-424-02	22.0	6.0	22	504	TRUE	FALSE
71	XXXXXXXX	0.0	0.0	0	0	TRUE	FALSE
72	XXXXXXXX	0.0	0.0	0	0	TRUE	FALSE
73	8-729-424-02	22.0	6.0	22	504	FALSE	FALSE
74	XXXXXXXX	0.0	0.0	0	0	TRUE	FALSE
75	XXXXXXXX	0.0	0.0	0	0	TRUE	FALSE
76	XXXXXXXX	0.0	0.0	0	0	TRUE	FALSE

Figura 4.11 – Seqüência de montagem final (ré-otimizada) da heurística 2

4.3.2 Comparações com outros otimizadores

A tabela 4.8, exibe os valores comparativos entre os critérios mencionados no item 4.2.5. Nesta tabela, as duas primeiras linhas reportam-se ao otimizador HLC, sendo que na primeira linha apenas houve a repetição dos resultados analisados em 4.2.5. Porém, na segunda linha, o dado mais importante trata-se do valor obtido para o tempo de montagem final, TM, o qual registrou um valor cronometrado médio de 23,9 segundos. No entanto, diferente do que aconteceu com na otimização do HLC exibido na primeira linha, o *estado* de partida para a otimização foi o próprio resultado obtido para a heurístaca 2 (exibidos na terceira linha da tabela 4.8). Conforme pode ser visto na tabela, desta vez o HLC não realizou trocas de *ventosas* e conseqüentes visitas ao ATC, como vinha ocorrendo quando o ponto de partida era a seqüência apresentada na figura 4.7. Este fato denota uma dependência acentuada daquele algoritmo com relação a seqüência inicial apresentada para otimização. Ademais, percebe-se que o HLC continuou extremamente rápido no sentido de encontrar uma solução considerada satisfatória. Também é interessante observar que o HLC manteve os 10 ciclos de montagem com apenas um componente (semelhante a heurística 2), como pode ser visto na coluna QC1Cap.

Tabela 4.8 – comparação entre as soluções obtidas pela heurística 2 , HLC , AG e AM

Método	QCG	QC4Cap	QC3Cap	QC2Cap	QC1Cap	QTVS	QVATC	TM	TP	TV
HLC	16	3	0	10	6	1	1	22,91s	~30s	1,98s
HLC reprocessado	-	-	-	-	10	0	0	23,9s	~30s	0
Heurística2 reotimizado	**19**	**0**	**4**	**2**	**10**	**0**	**0**	**22,35s**	**~1500s**	**0**
AG	12	0	2	8	0	3	1	29,3s	~1560s	-
AM	12	0	0	1	0	2	1	29,5s	-	-

A quarta e a quinta linhas da tabela 4.8, dizem respeito aos dados extraídos de trabalhos anteriores a este, os quais tinham o mesmo objetivo, mas enfocaram outras propostas. O primeiro trabalho foi elaborado por Craveiro, Carlos, 2005. O qual se

baseou no modelo de um algoritmo genético, AG. O segundo trabalho foi elaborado por Carvalho, Elidelson, 2007. O qual envolveu o uso de um algoritmo memético, AM.

Pois bem, a comparação dos tempos de montagem do GA com a heurística 2, referente a coluna TM da tabela acima, indica que houve um ganho substancial de tempo a favor da heurística 2. A economia de tempo foi da ordem de **23,72%.** As principais razões para isso, devem-se ao aumento de ciclos de montagem com apenas um componente e a eliminação da necessidade de trocas de ventosas, para a montagem dos mesmos componentes e utilizando-se o mesmo modelo de máquina de inserção. Apesar de parecer paradoxal aumentar-se a quantidade de ciclos de captura & montagem, de 12 para 19 ciclos, conforme exibido na coluna QCG, deve-se observar que tais ciclos montam somente um componente e todos os ciclos foram empurrados para a parte final da montagem. Este fenômeno garantiu uma maior velocidade montagem, além é claro, de ser a razão principal para a eliminação da troca de *ventosas*. Também deve ser registrado que houve uma discrepância entre o valor TM simulado, exposto na figura 4.12, o qual foi de **20,51** segundos contra um valor real cronometrado de **22,35** segundos. A diferença de 1,84 segundos foi atribuída às imperfeições relativas ao método de cronometragem média manual e valores de escala referentes a velocidade do braço e tempos de captura, inserção e giro dos componentes. Entretanto, a principal razão deve ser atribuída ao fato da máquina não ter acompanhado a seqüência de captura determinada pela heurística em alguns ciclos de captura. Por exemplo, no segundo ciclo exibido na figura 4.11, a seqüência de captura indicada pelo método **heurístico 2** foi: Primeiro capture simultaneamente os componentes que se encontra nos *escaninhos* 16 e 18, depois capture o componente que se encontra no *escaninho* **26**, e finalmente capture o componente que se encontra no *escaninho* **38**. Todavia, a máquina realizou a seguinte seqüência: Primeiro capturou simultaneamente os componentes dos *escaninhos* 16 e 18, depois capturou o componente que se encontrava no escaninho 38, para finalmente capturar o componente do *escaninho* 26. Ora, esta última representa uma ineficiência, pois foi necessário deslocar-se dos *escaninhos* 16/18 para o *escaninho* 38 e voltar ao *escaninho* 26 para somente aí realizar a captura, ou seja, esta volta seria desnecessária caso a máquina tivesse captura o componente que se encontrava no *escaninho* 26 no momento em que o braço se dirigia para o *escaninho* 38. Desta forma houve um acréscimo no tempo total de montagem real. O motivo pelo qual a máquina não dissociou a seqüência cíclica de

montagem da seqüência cíclica de captura ainda é desconhecido. O corpo técnico que acompanhou o teste também não soube configurar a máquina no sentido de realizar tal dissociação. Fica para o futuro investigar tal fenômeno posto que não há explicação lógica para tal dependência. Afinal, teoricamente, as cabeças de inserção podem capturar os componentes em uma seqüência e montá-los em outra.

Resta abortar o tempo de processamento (TP) até a obtenção das soluções propostas por AG e pela heurística 2, posto que não há como realizar comparações entre os tempos envolvidos durante as operações de troca de *ventosas* (TV), uma vez que não se dispõe deste dado para o AG.

Já foi dito que resultados de simulações obtidos da mesma forma dos ilustrados no item 4.2.3, conduzem ao fato haver convergência com um percentual reduzido de épocas/iterações, ou seja, com uma quantidade bem menor do que os 12.500.000 *estados* analisados até que o algoritmo da heurística 2 parasse e registrasse a solução exposta na figura 4.11. Então, seria necessário um tempo bem menor do que os 1500 segundos (25 minutos) que foram necessários para se chegar à solução presente. De qualquer forma, o tempo de processamento, mesmo sem qualquer otimizações que se poderiam impor ao código *C* elaborado em relação a estruturas de laços e de comparações, ficou abaixo do tempo reportado pelo AG em sua melhor solução... A redução foi de **3,8%** a favor da heurística 2. Contudo, como se trata de um valor percentual reduzido, em outras simulações poderia haver uma inversão destes resultados, caso a ordem de combinações envolvidas na heurística 2 se mantenha nos mesmos patamares... Ademais esta comparação só estaria completa se houvesse condições de traçar uma relação entre a quantidade de gerações e populações envolvidas no algoritmo genético abordado e a quantidade de *estados* envolvidos na heurística 2.

Por outro lado, de forma bastante objetiva, observa-se no gráfico da figura 4.12, que a convergência para a solução em **20,51** segundos, ocorreu no reinício de número **1211**, o que corresponde a um percentual de **48%** iniciais em relação aos **2500** reinícios antes da parada. Desta forma, poder-se-ia estimar (em função do caráter aleatório do método) que a solução foi encontrada, no mínimo, na metade do tempo de processamento

indicado na tabela 4.8. Portanto, o tempo cairia de 1500 segundos para pouco menos 750 segundos.

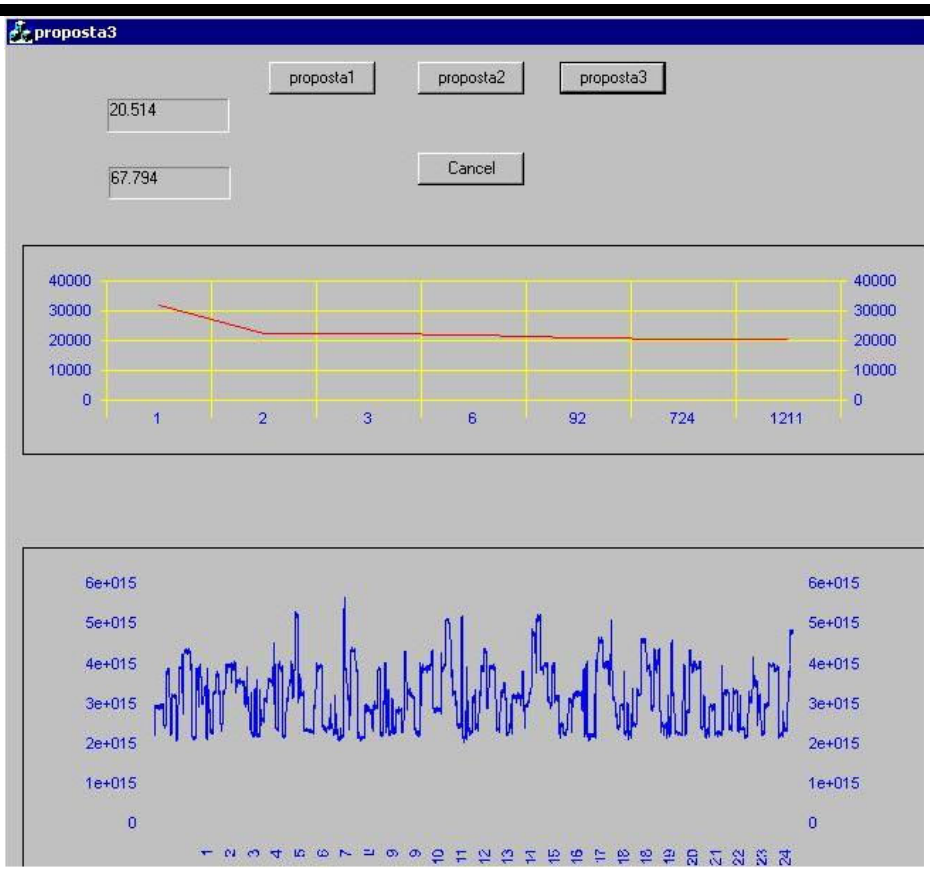

Figura 4.12 – performance para 44 componentes em 2500x5000 iterações para a Heurística 2 modificada

Resta tecer alguns comentários comparativos entre a linha com dados sobre o algoritmo memético, AM, extraído do trabalho realizado por Carvalho, Elidelson, 2007 já citado em ocasiões anteriores, e a heurística 2 eleita no trabalho presente.

Deve ser esclarecido que os critérios de avaliação daquele trabalho parecem não coincidir com os definidos aqui! Por exemplo, na heurística 2 considera-se que o critério QC4Cap só é realmente simultâneo, *se e somente se*, os *alimentadores* nos quais se encontram os componentes a serem capturados, estiverem rigorosamente alinhados com as respectivas cabeças de inserção e suas *ventosas*. Ou seja, caso as

cabeças h1-h2-h3-h4 capturem, ao mesmo tempo, os componentes inerentes em um dado ciclo, então os alimentadores devem possuir posições adjacentes e crescentes. Neste caso, se a cabeça h1 estiver posicionada sobre o alimentador 1, então a cabeça h2 deve estar posicionada sobre o alimentador 3; a cabeça h3 sobre o alimentador 5, deixando h4 com o alimentador 7. Critérios equivalentes foram aplicados a QC3Cap, QC2Cap e QC1Cap...

Além disso, caso não haja capturas simultâneas no mesmo ciclo, é porque ele conteria quatro, três, dois ou um componente capturado em tempos distintos com distintos deslocamentos do braço.

A figura 4.13 ilustra o exemplo anterior, onde se deve observar que a cabeça h3 quebra a seqüência de captura simultânea, pois a mesma, deve deslocar todo o conjunto, para que seja possível capturar o componente que se encontra no alimentador 11. Ou seja, haveria captura de apenas três componentes simultâneos, através das cabeças h1, h2 e h4. Após o referido deslocamento, a cabeça h3 pegaria o último componente do ciclo.

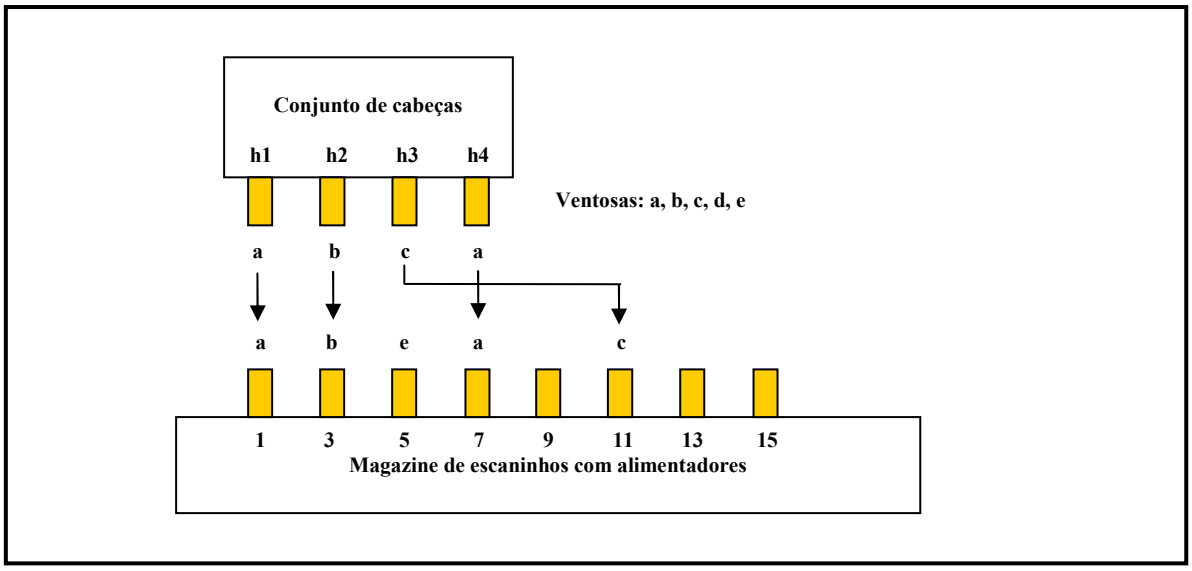

Figura 4.13 – ciclo com captura de somente 3 componentes simultaneamente nos alimentadores 1, 3 e 5; Depois há um deslocamento do conjunto para permitir h3 capturar o componente no alimentador 11.

Com isso em mente, refizeram-se os valores para os critérios da última linha da tabela 4.8 estabelecendo a comparação com as mesmas definições para os critérios utilizados. De qualquer forma, e também por isso, limitar-se-á a comparação ao nível do critério tempo de montagem (TM). Como se pode observar na tabela, o melhor TM alcançado com um algoritmo memético (AM) utilizando têmpera simulada como processo de busca local, foi de 29,5 segundos para uma otimização com os mesmos 44 componentes. Portanto, a heurística 2 gerou um ganho da ordem de **24,4%** uma vez que consegue montar os componentes em apenas 22,3 segundos. Como não se dispõe da informação relativa ao tempo de processamento do AM, não há como realizar comparação entre os métodos relativa a este critério.

5. Conclusões

A pesquisa demonstrou a eficácia das heurísticas propostas (principalmente a heurística 2, pois foi a escolhida para aprofundar a avaliação) associadas ao algoritmo de subida de encosta e o poder da técnica de reinícios aleatórios de *estados*. Contudo, as simulações realizadas durante este trabalho não demonstraram qualquer vantagem significativa da heurística 2 sobre a heurística 1, ou vice-versa. Porém, testes mais aprofundados deverão ser realizados de forma a ratificar ou não esta tendência. Por exemplo, em trabalhos futuros, poder-se-ia variar a quantidade de cabeças que compõem um ciclo (no trabalho presente, a quantidade foi fixada em quatro cabeças) e observar se uma das heurísticas se sobrepõe à outra...

Tanto os resultados simulados quanto os resultados práticos ratificaram que o método de subida em encosta com reinício aleatório, é capaz de solucionar, com vantagens, o problema de otimização da seqüência de montagem de componentes eletrônicos sobre placas de circuito impresso por máquinas de inserção automática. As vantagens dizem respeito tanto à capacidade de redução do tempo de montagem dos componentes, quanto do tempo de convergência para alcançar a solução. Sendo assim, tanto o objetivo geral quanto os objetivos específicos dessa dissertação foram atingidos.

O algoritmo criado foi capaz de explorar gigantescos espaços de busca variando-se seus parâmetros o que permitiu observar a rapidez desta convergência. A simplicidade do algoritmo de subida de encosta foi quem tornou versátil as alterações de parâmetros mais sutis enriquecendo o experimento. Um exemplo desta sutileza disse respeito ao controle do número de trocas de *ventosas* em um mesmo ciclo e do envolvimento obrigatório ou não dos elementos que compõem o primeiro ciclo durante o processo de formação de novos *estados*. Tais experimentos foram fundamentais para se alterar os critérios que conduzem as melhores seqüências de montagem (aquelas, com tempo de montagem reduzido). Foi surpreendente observar que este algoritmo reduziu o tempo de montagem de uma seqüência com 44 componentes da ordem de 28 segundos (para soluções sugeridas por algoritmos do tipo AG ou AM) para algo em torno de 22

segundos, baseando-se naquelas experiências. Isso representa um ganho específico de **21,4%** no tempo para montar tais componentes.

Em última análise os resultados práticos conduziram a um tempo de montagem de 22,3 segundos contra 23,9 segundos da solução do algoritmo do fabricante, o HLC. Ou seja, houve um ganho de pelo menos 6,6% na montagem da placa com os mesmos 44 componentes usados como teste principal.

A despeito de pequenas imprecisões oriundas das coordenas de referencia ou velocidades de braços envolvidas no cálculo do tempo de montagem de componentes, o simulador respondeu computando um tempo compatível com os resultados práticos cronometrados. Desta forma, seria possível aperfeiçoá-lo na forma de uma ferramenta comercial de análise de inserção de componentes...

Por outro lado, restou a sensação de que muitos experimentos ainda poderiam ser explorados, dado a versatilidade para combinação de parâmetros, afinal, o código foi integralmente elaborado durante o decurso da pesquisa, o que permite o total controle sobre o que esta acontecendo. Algumas destas experiências já foram realizadas. Porém, a realização de novas experiências poderá conduzir a otimizações cada vez mais rápidas, precisas e direcionadas...

6. Trabalhos futuros

No último parágrafo do capítulo anterior foi dito que muitos outros experimentos poderiam ser realizados na busca por resultados cada vez melhores. Dentre tais experiências, poder-se-iam destacar:

a) Alimentar os algoritmos com valores precisos de velocidade de deslocamento de braço e tempo inicial para ir do ATC até o magazine.

b) Realizar testes com outros modelos de máquinas insersoras e aumentar gradativamente a quantidade de componentes a otimizar;

c) Otimizar o código fonte para torná-lo mais rápido e robusto; Investigar qual a melhor estratégia para acelerar ao máximo a convergência da solução;

d) Realizar um quarto nível de otimização referente à distribuição das *ventosas* sobre o dispositivo de ATC em função da quantidade de componentes diferentes a serem montados;

e) Definir, de forma clara, um critério para limitar seqüências de ciclos com cabeças vazias, e desta forma, forçando visitas ao ATC, se for o caso;

f) Realizar experimentos para avaliar a gradual substituição do caráter aleatório na formação dos *estados* iniciais por processos mais determinísticos e, desta forma, menos sujeito a testes repetidos;

g) Testar outros métodos de busca, como por exemplo, o método de têmpera simulada (*simulated annealing*) aplicado à solução do problema de otimização ora exposto;

h) Aprofundar a avaliação sobre a heurística 1 no sentido de especificar se existe alguma vantagem ou desvantagem pontual deste método em relação a heurística 2.

7. Referências bibliográficas

Areibi, S. *Ph.D Thesis: Towards Optimal Circuits Layout Using Advanced. Seach Techniques,*1995.

Areibi, S., Vannelli, A. *Circuit Partitioning Using a Tabu Search Approach. IEEE International Symposium on Circuits and Systems*, pp. *1643–1646, Chicago, Illinois, 1993.*

Areibi, S. *An Integrated Genetic AlgorithmWith Dynamic Hill Climbing for VLSI Circuit Partitioning. Genetic and Evolutionary Computation Conference (GECCO-2000), Las Vegas, Nevada, July 2000, IEEE.*

Ayob, M., Kendall, G. *Real-time Scheduling for Multi Headed Placement Machine. Proceeding of the 5th IEEE International Symposium on Assembly and Task Planning, Besançon, France, July 10-11,2003*.

Backer, B., Furnon, V., Kilby P., Prosser, P., Shaw, P. *Solving Vehicle Routing Problems Using Constraint Programming and Metaheuristics. Journal of Heuristics, 6: 501–523 (2000), Kluwer Academic Publishers.*

Bai, R., Kendall, G. *An Investigation of Automated Planograms Using a Simulated Annealing Based Hyper-Heuristics. Proceedings of the Fifth Metaheuristics International Conference*, *Kyoto, 2003.*

Baluja, S., Caruana, R. *Removing the Genetics from the Standard Genetic Algorithm.* Prieditis, A., Russell, S., *editors. Proceedings of the International Conference on Machine Learning*, pages 38–46. Morgan Kaufmann, *1995*.

Ball, M. *Sequencing of Insertion in Printed Circuit Board Assembly. Operations Research,* v. 36, n. 2, pp.192-210, 1989.

Burke, E. K., Cowling, P. I., Keuthen, R. *New Models and Heuristics for Component Placement in Printed Circuit Board Assembly. Proceedings of the 1999 International Conference on Information Intelligence and Systems, pp.133-140, 1999*.

Burke, E. K., Cowling, P. I., Keuthen, R. *The Printed Circuit Board Assembly Problem: Heuristics Approach for Multi-Head Placement Machinery. International Conference on Artificial Intelligence. 2001*.
http://www.asap.cs.nott.ac.uk/publications/ps/ralf_PCBAP.ps

Carvalho, E. Uso de Algoritmos Meméticos na Utilização de Seqüências de Montagem de Máquinas SMD. Tese de Mestrado, UFAM, 2007.

Craveiro, C. H. Otimização do Posicionamento e da Montagem Automática de Componentes Eletrônicos Através de um Algoritmo Genético. Tese de mestrado, COPPE, UFRJ, 2005.

Feo, T., Resende, M., Smith, S. *A Greedy Randomized Adaptive Search Procedure for The Maximum Independent Set*, *Operations Research*, 1994.

HO, W., JI, P. *A Hybrid Genetic Algorithm for Component Sequencing and Feeder Arrangement. Journal of Intelligent Algorithm, v.15, pp.307- 315, 2004.*

Kernighan, B.W., Lin, S. *An Efficient Heuristic Procedure for Partitioning Graphs. The Bell System Technical Journal, vol. 49, n. 2, pp. 291–307, February 1970.*

Lee, W., Lee, S., Lee, B., e Lee, Y. *A Genetic Optimization Approach to Operation of a Multi Head Surface Machine*. *IEICE Trans.Fundamentals*, *v. E83-A, n. 9, pp. 1748-1756, Sep. 2000.*

Leu, M. C., Wong, H., Ji, Z. *Planning of component placement/insertion sequence and feeder setup in PCB assembly using genetic algorithm. Journal of Electronic Packaging, Transactions ASME, 115, pp. 424–432, 1993.*

Maschio, C., Schiozer, D. J. *Ajuste de Histórico Assistido Usando Métodos de Otimização de Busca Direta.* IBP06204, *Anais do Rio Oil & Gas Conference*, Rio de Janeiro, 4-7 de Outubro, 2004.

Mendes, A. S. Algoritmos Meméticos Aplicados aos Problemas de Sequenciamento em Máquinas. Tese de mestrado, Unicamp,1999.

Rabak, C. S. Otimização do processo de inserção automática de componentes eletrônicos empregando a técnica de *Times* Assíncronos. Escola Politécnica da Universidade de São Paulo, 1999.

Russel, S. Inteligência Artificial, pp.67- 147.

Sucupira, I. R. Um Estudo Empírico de Hiper-Heurísticas. Instituto de Matemática e Estatística, Universidade de São Paulo, 2007.

Wang, C., Ho, L.H., Fu, H.I. e Su, Y.C. *A magazine assignment heuristic for robotic assembly using the dynamic pick-and-place approach. International journal of industrial Engineering, Vol 4, No. 1, pp 24-33*, 1997.

Zeng, Y. *Hierarchical Planning for a Surface Mounting Machine Placement***.** *Journal of Zhejiang University, 2004*.

8 Anexo

Neste anexo será apresentado um resumo de simulações realizadas com a heurística 2, após implementadas as otimizações descritas no capítulo 3 (Metodologia). Nas tabelas A.1 a A.23, encontram-se os melhores resultados obtidos a partir de tais simulações. Ou seja, não se optou por mostrar resultados médios, e sim os melhores resultados segundo o critério de tempo de montagem. Portanto, os tempos de processamento apenas acompanham aquele critério. Por outro lado, poder-se-ia ter realizado simulações elegendo-se o tempo de processamento como critério. Em ambos os casos, os melhores resultados são aqueles que minimizam os valores. Finalmente cabe enfatizar que o caráter de aleatoriedade do método tendeu a não privilegiar um critério sobre o outro, ou seja, encontraram-se bons resultados tanto com pouco quanto com muito tempo de processamento. Também se deve ater para o fato de que quanto maior for à quantidade de reinícios dos laços de iterações, maior será o tempo de processamento.

Os componentes e respectivas coordenadas sobre PCI são os mesmos listados na figura 4.3, capítulo resultados obtidos, na página 84.

Nos títulos de cada tabela exibida a seguir, encontram-se os seguintes termos:

1) Repete Rolo – RR – significa que há mais de um escaninho-alimentador com o mesmo tipo de componentes;
2) Não Repete Rolo – NRR – significa que não há nenhum escaninho-alimentador alocando um tipo de componente que já está alocado em outro escaninho-alimentador;
3) Trocax1 – T1 – significa que haverá no máximo uma troca de ventosa em cada visita ao ATC;
4) Trocax2 – T2 – significa que haverá no máximo duas trocas de ventosas em cada visita ao ATC;
5) Trocax3 – T3 – significa que haverá no máximo três trocas de ventosas em cada visita ao ATC;

6) Trocax4 – T4 – significa que haverá no máximo quatro trocas de ventosas em cada visita ao ATC;
7) Mudar Escaninhos – ME – significa que escaninhos novos podem ser aleatoriamente selecionados com relação a seqüência de montagem inicial;
8) Não Mudar escaninhos – NME – significa que nenhum novo escaninho será escolhido, ou seja, só haverá permutas entre os escaninhos alocados no início da simulação;
9) Lim1 = xxxx – significa que à variável de parada do algoritmo, Limite 1, será atribuído um valor xxxx;
10) Lim2 = xxxx – significa que à variável de parada do algoritmo, Limite 2, será atribuído um valor xxxx;
11) Mesmo início – Mi – significa que todas as simulações de uma determinada série, iniciarão a partir da mesma seqüência de montagem inicial;
12) Reinício – Re – significa que, em uma série de simulações de um determinado tipo, a seqüência inicial da atual será a seqüência final da simulação anterior.
13) Permuta xY – PY – significa que tanto os componentes na seqüência de montagem, quanto os tipos de componentes sobre os escaninhos-alimentadores serão permutados Y vezes e de forma aleatória (respectivamente em suas seqüências).

8.1 Tabelas NRR

Tabela A.1 – NRR; T1; P1; ME; lim1 = lim2 = 100; Mi.

Slots	Ciclos	Tempo de montagem
2~32	19	18,32
2~32	19	18,68
2~32	19	20,85
2~32	19	18,71
2~32	17	19,66
2~60	17	24,36
2~60	19	24,85
2~60	18	25,72
2~60	22	34,02
2~60	18	24,33
22~60	19	30,71
22~60	17	27,25
22~60	18	28,96
22~60	17	27,66
22~60	17	27,61
30~60	17	29,51
30~60	18	30,62
30~60	18	31,76
30~60	18	32,33
30~60	18	32,89

Tabela A.2 – NRR; T1; P4; escaninho 2~60; Lim1 varia; Lim2=100; Mi.

Limite1	Ciclos	Tempo de montagem	Tempo de processamento
100	17	22,95	4,3
200	19	23,79	8,51
300	19	24,36	16,67
500	18	24,57	16,32
900	19	25,22	36,29
1000	19	23,00	44,14

Tabela A.3 – NRR; T1; P4; escaninho 2~60; Lim2 varia; Lim1=100; Mi.

Limite2	Ciclos	Tempo de montagem	Tempo de processamento
100	18	25,00	4,21
200	18	23,94	8,12
300	17	23,50	16,37
500	17	22,00	27,09
900	18	21,98	36,72
1000	17	21,83	41,19

Tabela A.4 – NRR; T1; P8; escaninho 2~60; Lim1 varia; Lim2=100; Mi.

Limite 1	Ciclos	Tempo de montagem	Tempo de processo
100	17	24,46	4,17
200	17	24,12	8,31
500	17	24,16	21,34
1000	19	23,47	41,44

Tabela A.5 – NRR; T1; P8; escaninho 2~60; Lim2 varia; Lim1=100; Mi.

Limite 2	Ciclos	Tempo de montagem	Tempo de processamento
100	18	24,03	5,18
200	19	22,74	8,06
500	19	22,60	20,17
1000	17	22,72	44,03

Tabela A.6 – NRR; T1; P12; escaninho 2~60; Lim1 varia; Lim2=100; Mi.

Limite 1	Ciclos	Tempo de montagem	Tempo de processamento
100	17	24,07	4,90
500	18	24,59	20,70
1000	18	23,32	41,08

Tabela A.7 – NRR; T1; P12; escaninho 2~60; lim2 varia; Lim1=100; Mi.

Limite 2	Ciclos	Tempo de montagem	Tempo de processamento
100	17	24,20	4,8
500	19	23,11	21,25
1000	18	23,56	41,82

8.2 Tabelas NRR

Tabela A.8 – RR; T1; P1; Lim1 = Lim2=100; Mi.

Slots	Ciclos	Tempo de montagem
2~32	19	24,08
16~50	18	23,51
2~60	18	24,78

Tabela A.10 – RR; T2; P1; escaninho 2~60; Mi.

Limite1	Limite 2	Ciclos	Tempo de montagem	Tempo de processamento
100	100	14	24,19	4,01
1000	1000	13	22,23	6:43,7

Tabela A.11 – RR; T3; P1; escaninho 2~60; Mi.

Limite1	Limite 2	Ciclos	Tempo de montagem	Tempo de processamento
100	100	12	24,29	4,04
1000	1000	12	24,94	6:35,32

Tabela A.12 – RR; T4; P1; escaninho 2~60; Mi.

Limite 1	Limite 2	Ciclos	Tempo de montagem	Tempo de processamento
100	100	12	27,75	4,23
100	100	12	28,34	2,72
1000	1000	13	26,01	6:38,16

Tabela A.13 – RR; T1; P4; ME; Lim1= Lim2=100; Mi.

Slots	Ciclos	Tempo de montagem
2~32	18	20,43
2~32	19	21,54
2~32	18	20,56
2~32	18	20,31
2~32	17	19,89
2~60	19	26,12
2~60	18	24,25
2~60	17	24,25
2~60	18	27,61
2~60	17	23,09
22~60	19	28,37
22~60	17	26,09
22~60	19	28,88
22~60	19	29,44
22~60	18	27,56

Tabela A.14 – RR; T1; P1; escaninho 2~60; `Lim1 varia; Lim2=100; Mi.

Limite 1	Limite 2	Nº de Ciclos	Tempo de montagem	Tempo de processamento
100	100	17	23,33	5,69
200	100	18	23,03	7,76
300	100	17	21,38	14,09
400	100	17	22,46	16,13
500	100	19	21,97	24,17
600	100	17	21,9	27,31
700	100	17	22,68	29,19
800	100	17	22,24	38,46
900	100	17	22,51	39,06
1000	100	17	22,62	40,28

Tabela A.15 – RR; T1; P1; escaninho 2~60; Lim2 varia; Lim1=100; Mi.

Limite 1	Limite 2	Nº de Ciclos	Tempo de montagem	Tempo de processamento
100	100	17	23,33	5,69
100	200	19	22,18	8,11
100	300	17	21,64	13,26
100	400	19	21,90	16,43
100	500	19	22,08	23,61
100	600	19	21,45	25,15
100	700	19	21,3	31,28
100	800	19	21,90	37,80
100	900	17	22,07	43,95
100	1000	19	21,52	46,11

Tabela A.16 – RR; T1; P4; escaninho 2~60; Lim1 varia; Lim2=100; Mi.

Limite 1	Limite 2	Nº de Ciclos	Tempo de montagem	Tempo de processo
100	100	17	22,38	4,43
200	100	17	21,96	10,62
300	100	19	21,62	14,29
500	100	18	22,49	22,59
900	100	17	22,1	43,66
1000	100	17	21,86	44,08

Tabela A.17 – RR; T1; P4; escaninho 2~60; Lim2 varia; Lim1=100; Mi.

Limite 1	Limite 2	Nº de Ciclos	Tempo de montagem	Tempo de processamento
100	100	19	22,11	4,85
100	200	19	21,86	8,82
100	300	17	22,32	13,79
100	500	19	21,60	22,77
100	900	17	22,00	41,37
100	1000	19	21,09	44,99

Tabela A.18 – RR; T1; P8; escaninho 2~60; Lim1 varia; Lim2=100; Mi.

Limite 1	Limite 2	Nº de Ciclos	Tempo de montagem	Tempo de processamento
100	100	17	22,13	4,56
200	100	17	22,35	9,16
500	100	19	21,35	24,26
1000	100	17	21,42	44,89

Tabela A.19 – RR; T1; P8; escaninho 2~60; Lim2 varia; Lim1=100; Mi.

Limite 1	Limite 2	Nº de Ciclos	Tempo de montagem	Tempo de processamento
100	500	19	22,23	22,95
100	1000	18	21,46	44,27

Tabela A.20 – RR; T1; P12; escaninho 2~60; Lim1 varia; Lim2=100; Mi.

Limite 1	Limite 2	Nº de Ciclos	Tempo de montagem	Tempo de processamento
100	100	17	22,33	4,73
500	100	19	22,06	23,46
1000	100	19	21,46	45,04

Tabela A.21 – RR; T1; P12; escaninho 2~60; Lim2 varia; Lim1=100; Mi.

Limite 1	Limite 2	Nº de Ciclos	Tempo de montagem	Tempo de processamento
100	500	19	21,83	23,38
100	1000	17	21,68	44,73
100	2000	19	20,95	1:30,8
100	3000	19	20,82	2:15,11

Tabela A.22 – RR; T1; P1; escaninho 2~60; Mi.

Limite 1	Limite 2	Nº de Ciclos	Tempo de montagem	Tempo de processamento
100	100	18	22,80	20,66
1000	1000	19	22,30	7:47,25

Tabela A.23 – RR; T1; P1; slots 2~60; Lim1= Lim2=100; Inicia a partir do anterior.

Início	Nº de Ciclos	Tempo de Montagem	Inicia do anterior
1	19	25,99	-
2	19	25,98	1
3	17	23,81	2
4	18	25,09	3
5	19	25,64	4
6	19	25,72	5
7	19	25,67	6
8	19	25,99	7
9	19	25,76	8
10	18	25,95	9

As tabelas anteriores foram formadas a partir de séries de simulações com a intenção de investigar as seguintes situações:

1) Variação de desempenho com os parâmetros Lim1, Lim2 e número de posições permutadas com relação à seqüência de montagem e seqüência de escaninhos-alimentadores;
2) Minimização quanto ao número de troca de ventosas nas visitas ao ATC e quanto ao número de ciclos de montagem;
3) Repetição de rolos (escaninhos-alimentadores) contendo o mesmo tipo de componente;
4) Influencia da "inicialização", ou seja, da distribuição aleatória inicial;
5) Influencia de posição dos alimentadores;
6) Influencia de aumentar ciclos com relação ao número de trocas de ventosas e vice-versa;

Conforme mencionado no início deste anexo, A tabela A.24, representa a extração dos melhores resultados, segundo o critério de minimização do tempo de montagem, das tabelas A.1 a A.23, lembrando que aquelas tabelas representam um resumo do comportamento de uma quantidade bem maior de simulações segundo as condições definidas em seus títulos, os quais foram resumidos na primeira coluna da tabela A24.

Tabela A.24 – Melhores resultados segundo o critério de tempo de montagem.

Item	Configurações	Tempo de montagem em segundos	Tempo de processamento em segundos	Número de ciclos de captura e monta
1	NRR-T1-P1-Mi-Lim1=Lim2=100-escaninho 2~32	18,32	4,1	19
2	NRR-T1-P1-Mi-Lim1=Lim2=100-escaninho 2~60	24,33	4,1	18
3	NRR-T1-P1-Mi-Lim1=Lim2=100-escaninho 22~60	27,25	4,1	17
4	NRR-T1-P1-Mi-Lim1=Lim2=100-escaninho 30~60	29,51	4,1	17
5	NRR-T1-P4-Mi-Lim1=Lim2=100-Lim1 varia	22,95	4,31	17
6	NRR-T1-P4-Mi-Lim2 varia	21,83	41,19	17
7	RR-T2-P1-Mi-Lim1= Lim2=1000	22,23	403,7	13
8	RR-T3-P1-Mi-Lim1=Lim2=100	24,29	4,04	12
9	RR-T4-P1-Mi-Lim1= Lim2=1000	26,01	398,16	13
10	RR-T1-P1-Mi-Lim1=Lim2=100-escaninho 2~32	19,89	4,1	17
11	RR-T1-P1-Mi-Lim1=Lim2=100-escaninho 2~60	23,09	4,1	17
12	RR-T1-P1-Mi-Lim1=Lim2=100-escaninho 22~60	26,09	4,1	17
13	NRR-T1-P8-Mi-Lim1 varia	23,47	41,44	19
14	NRR-T1-P8-Mi-Lim2 varia	22,60	20,17	19
15	NRR-T1-P12-Mi-Lim1 varia	23,32	41,08	18
16	NRR-T1-P12-Mi-Lim2 varia	23,11	21,25	19
17	RR-T1-P1-Mi-Lim1= Lim2	22,8	20,66	18
18	RR-T1-P1-Mi-Lim1 varia	21,38	14,09	17
19	RR-T1-P1-Mi-Lim2 varia	21,3	31,28	19
20	RR-T1-P4-Mi-Lim1 varia	21,62	14,29	19
21	RR-T1-P4-Mi-Lim2 varia	21,09	44,99	19
22	RR-T1-P8-Mi-Lim1 varia	21,35	24,26	19

23	RR-T1-P8-Mi-Lim2 varia	22,23	22,95	19
24	RR-T1-P12-Mi-Lim1 varia	21,46	45,04	19
25	RR-T1-P12-Mi-Lim2 varia	20,82	135	19
26	RR-T1-P1-Ri-Lim1=Lim2	23,81	3º reinicio – vide Tabela A.23	17
27	RR-T1-P1-Mi-Lim1= Lim2	22,8	20,66	18

8.3 Gráficos construídos com os dados da tabela A.24

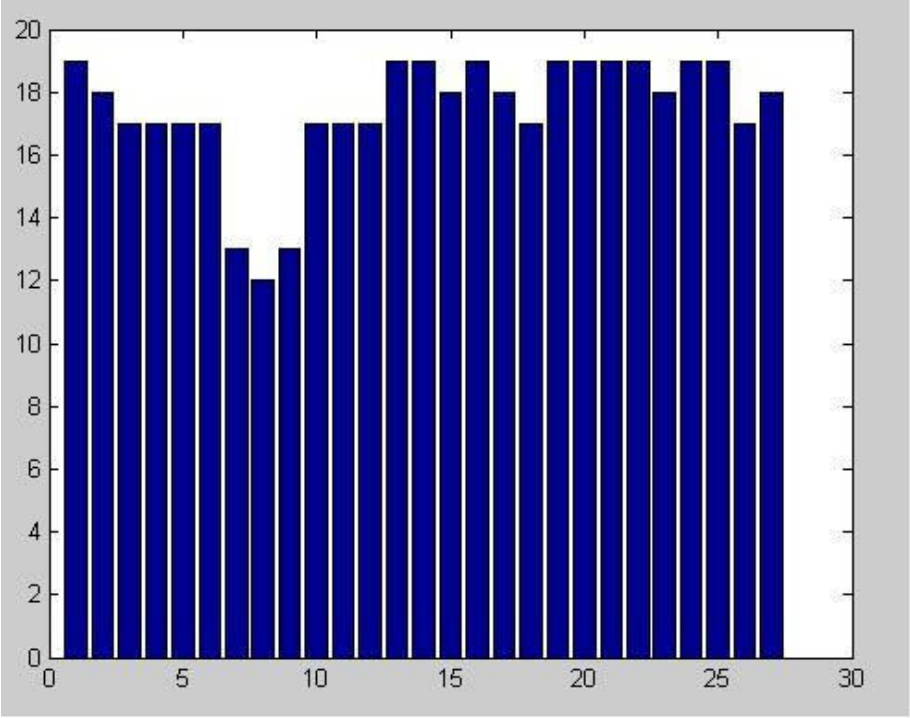

Figura A.1 – Itens da tabela A.24 x número de ciclos

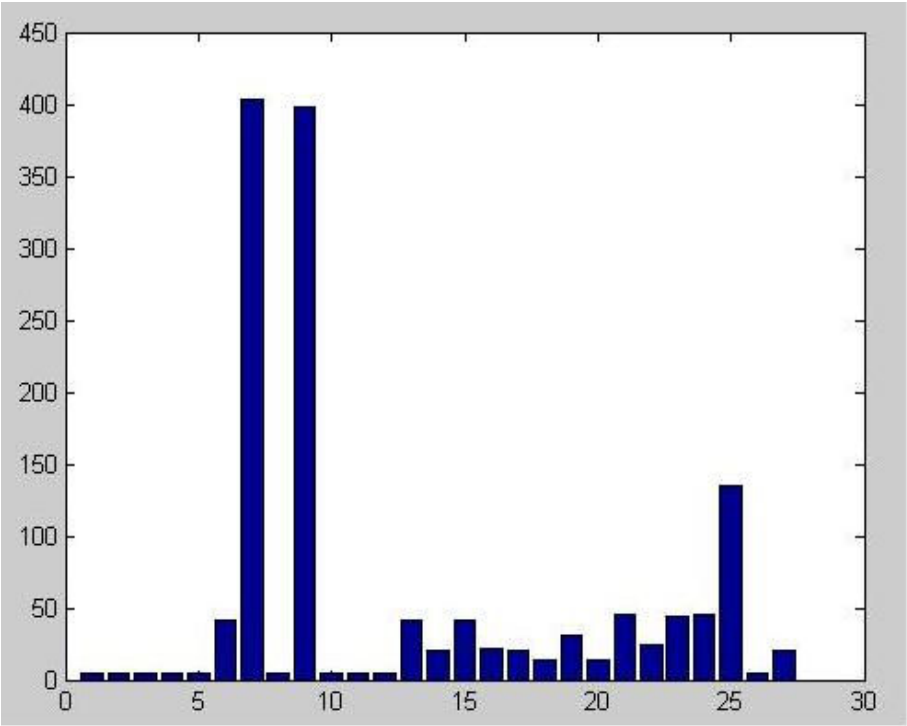

Figura A.2 – Itens da tabela A.24 x tempo de processamento em segundos

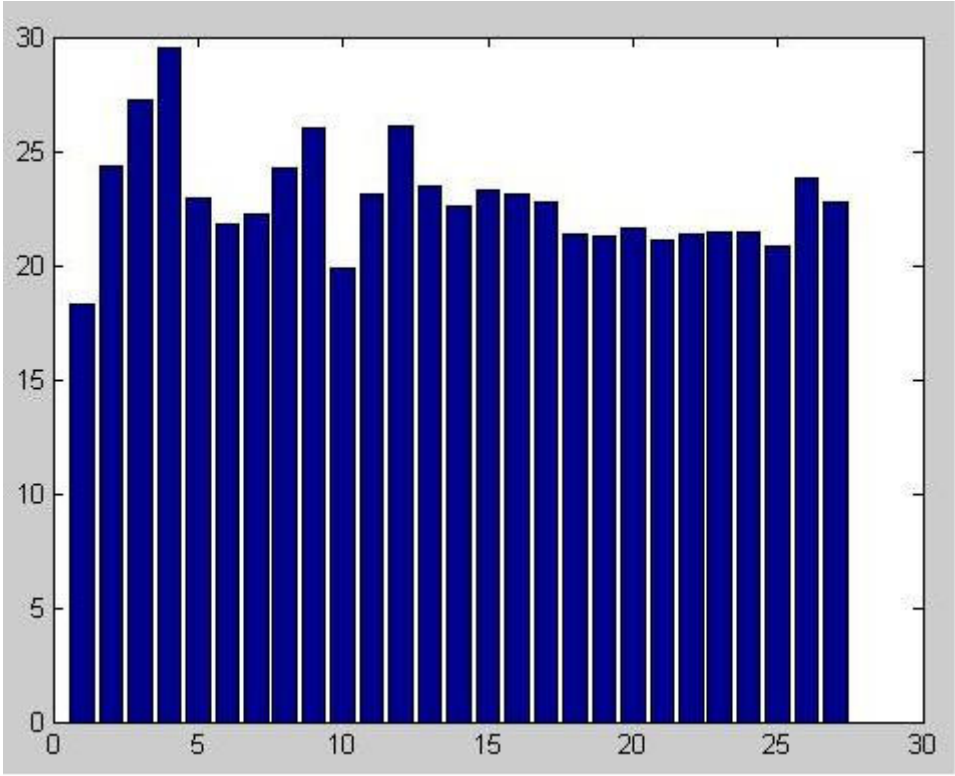

Figura A.3 – Itens da tabela A.24 x tempo de montagem em segundos

8.4 Observações extraídas das tabelas e dos gráficos anteriores

a) Com relação a tabela A.23 pode-se constatar que a melhor resposta em termos de tempo de montagem (23,81s) foi obtido na terceira iteração, a qual partiu de uma solução pior (25,98s) advinda da segunda iteração. Ou seja, obteve-se uma melhoria de resultado. Porém, a mesma situação não foi mantida na iteração seguinte. Pois, como se pode observar na quarta iteração, houve uma piora de desempenho ao se obter uma solução com 25,09s para o tempo de montagem. Desta forma, uma solução inicial pior pode gerar uma solução final melhor e vice-versa. Este comportamento é coerente com o fato de tratar-se de uma heurística que gera soluções aleatórias e avalia seu desempenho;

b) Com relação à tabela A.12 é interessante observar que ao aumentar-se drasticamente a quantidade de iterações tanto para o parâmetro Lim1 quanto para o parâmetro Lim2, em geral, provoca uma melhoria na solução final. Naquela tabela, por exemplo, quando Lim1=Lim2=100, o tempo de montagem foi de 28,34s. Contudo, ao aumentar-se Lim1=Lim2-1000, este tempo cai para 26,01s, representando um ganho de 8,22%. Mas apesar deste ganho que pode ser considerado como uma tendência, aumentar-se muito estes parâmetros provoca um aumento proporcional no tempo de processamento, pois neste caso, houve um aumento de 2,72s para aproximadamente 398s, ou seja, uma perda de 99,31%;

c) Tanto na tabela A.11 quanto na tabela A.13, obteve-se as melhores soluções ao restringir-se a faixa dos escaninhos que poderiam ser usados na formação da seqüência de montagem ao escaninho da posição 2 até o escaninho da posição 32. Tais escaninhos foram suficientes para comportar todos os tipos de componentes usados na PCI a ser montada. Ambas as tabelas exibem resultados para outras faixas de escaninhos para montar os mesmos tipos de componentes. Os melhores resultados ficaram em 18,32s e 19,89s, respectivamente para a tabela A.1 e A.13. No caso da tabela A.1, quando a faixa foi estendida para escaninhos de 2 a 60, a melhor solução ficou em 24,33s; com faixa de 22 a 60, a melhor solução ficou em 27,25s e quando a faixa foi restringida entre 30 a 60, a melhor solução ficou em 29,51s. Portanto, no mínimo, restringir-se a faixa de escaninhos entre 2 a 32, o ganho ficou em 24,7% (comparando 18,32s contra 24,33s). Esta mesma tendência foi observada para os dados da tabela A.13. Além disso, a solução de 18,32s foi a melhor dentre todas as soluções encontradas e exibidas nas tabelas A.1 a A.23. Tal comportamento indica haver um comprometimento entre a posição dos escaninhos quanto a posição da PCI e coordenadas dos componentes a serem montados;

d) Na tabela A.24 observou-se que uma quantidade de ciclos menor não tem como conseqüência direta, um menor tempo de montagem para a mesma PCI. Pois, no item 1 a quantidade de ciclos foi de 19, com tempo de montagem em 18,32s; já no item 3, foram 17 ciclos, com tempo de montagem em 27,25s; Além disso, no item 8, forma apenas 13 ciclos (conseqüência direta de optar-se por maximizar a troca de ventosas na visita ao ATC – para três neste caso), com tempo de montagem em 24,29s. Estes resultados confirmam o que foi dito no inicio deste parágrafo.

e) Observando a figura A.1, no item 8 obteve-se a menor quantidade de ciclos, 12, a qual foi conseqüência de uma configuração que trocava até três ventosas por visita ao ATC. Por outro lado, no item 9, a quantidade de ciclos ficou em 13, para uma configuração que trocava até quatro ventosas por visita ao ATC. Isso significa que não há garantia de minimizar a quantidade de ciclos quando se maximiza a quantidade de trocas de ventosas por visita ao ATC. Além disso, observando o item 1 da mesma tabela, onde se obteve a melhor solução geral para tempo de montagem (função objetivo), a quantidade de ciclos foi de 19. Ou seja, a solução com 19 ciclos ficou em 18,32s e a solução com 12 ciclos ficou em 24,29s. Então houve um ganho da ordem de 24,57% a favor de uma solução com uma quantidade de ciclos da ordem de 36,84% superior. Foi constatado que tal comportamento foi tendencioso, entretanto, isso não é necessariamente um comportamento geral, mas, talvez o seja, para o tipo de máquina usada;

f) Quanto figura A.2, nos itens 7,9 e 25 observou-se uma discrepância acentuada com relação ao tempo de processamento. Tal comportamento é uma conseqüência direta de aumentar-se significativamente os valores dos parâmetros Lim1 e Lim2. No caso dos itens 7 e 9, onde os tempos foram mais acentuados, aumentou-se bastante, ambos os limites. Contudo, no item 25, fez-se lim2=3000, mantendo-se Lim1=100; Por outro lado, cabe ressaltar, observando-se os resultados das tabelas em geral, que é possível obter-se bons resultados para tempo de montagem, mantendo-se os valores de Lim1 e Lim2 na casa dos 100 ou 200, onde se observa que o tempo de processamento para o computador usado ficou em torno dos 4 e 8 segundos respectivamente. Contudo, da mesma forma que isso só é possível devido ao caráter de aleatoriedade do método, não há garantias que bons tempos de montagem (boas soluções) serão obtidos rapidamente;

g) Na figura A.3, visualiza-se no item 1, que o melhor resultado geral dentre todas as simulações foi obtido restringindo-se o uso de escaninhos da posição 2 a 32 e configurado para NRR – Não Repetição de Rolo (alimentador) com o mesmo tipo de componente. Tal solução ficou em 18,32s. Entretanto, no item 10, obteve-se uma solução tão boa quanto a primeira, com valor de 19,89s. Esta última solução foi conseqüência da mesma restrição de escaninhos, porém com configuração RR – Repetição de Rolos por tipo de componente. Desta forma, não se detectou qualquer vantagem comparativa entre usar NRR ou RR;

h) Finalmente, durante as simulações realizadas, não se notou qualquer tipo de vantagem comparativa do parâmetro Lim1 sobre o parâmetro Lim2 e vice-versa. Da mesma forma, não foi detectada qualquer melhoria significativa ao aumentar-se a quantidade de permutações entre os componentes da seqüência atual durante o processamento do algoritmo, bem como o aumento das permutas entre os alimentadores encaixados sobre os escaninhos. Conforme se pode averiguar nos itens de 18 a 25 para uma configuração em RR e nos itens 2,5,6 e de 13 a 16 para uma configuração em NRR, ao aumentar-se as permutações de apenas 1, para 4, 8 ou 12, não se observou qualquer tipo de melhoria tendenciosa nas soluções obtidas da como, por exemplo, foi detectada ao se restringir as posições dos escaninhos que poderiam ser ocupados por componentes.

g) Extração de informações da simulação (com menor tempo de montagem) segundo os aspectos estabelecidos no capítulo 4 – pág. 99.

Tabela A.25 – Aspectos quantitativos e qualitativos segundo os critérios abordados

Método	QCG	QC4Cap	QC3Cap	QC2Cap	QC1Cap	QTVS	QVATC	TM	TP	TV
Heurística 2	19	0	0	6	10	0	0	18,32s	4,1s	0

A tabela anterior confirma a tendência do algoritmo a privilegiar ciclos com apenas um componente desde que o tempo de montagem seja minimizado. Neste caso em particular, existiram dez ciclos com apenas um componente. Tal comportamento ocorreu em função de evitar-se ao máximo a troca de mais de uma ventosa por visita ao ATC.

Apesar da tabela não apresentar ciclos com quatro ou três capturas simultâneas de componentes, a quantidade de ciclos com duas capturas foram seis e a quantidade de ciclos de captura com espaço de um *gap* (17 milímetros) foram de oito e mais um ciclo com uma captura de quase três simultâneos.

Finalmente caberia ressaltar que esta solução foi oriunda do item 1 da tabela A.24. onde apenas os escaninhos da posição 2 até a posição 32 entraram na simulação. Desta forma, os nove últimos ciclos, nos quais, apenas um componente foi capturado e montado, encontravam-se bastante próximos uns dos outros, principalmente em termos de posição de escaninhos.

Listagem com seqüência de **montagem simulada em 18,32s**

Seq	Compo name	X	Y	Feeder	Nozzle	Dummy	Troca
1	8-719-058-24	16.0	5.0	8	504	FALSE	FALSE
2	1-216-186-91	8.0	4.0	22	503	FALSE	FALSE
3	1-218-289-91	23.0	6.0	26	502	FALSE	FALSE
4	1-162-923-91	10.0	4.0	14	501	FALSE	FALSE
5	8-719-058-24	10.0	6.0	8	504	FALSE	FALSE
6	1-216-186-91	11.0	4.0	22	503	FALSE	FALSE
7	1-218-289-91	24.0	6.0	26	502	FALSE	FALSE
8	1-162-923-91	15.0	2.0	14	501	FALSE	FALSE
9	8-719-069-60	13.0	3.0	2	504	FALSE	FALSE
10	1-216-809-91	13.0	2.0	24	503	FALSE	FALSE
11	1-218-289-91	20.0	6.0	26	502	FALSE	FALSE
12	1-216-067-91	24.0	2.0	16	501	FALSE	FALSE
13	1-125-891-91	8.0	6.0	10	504	FALSE	FALSE
14	1-216-186-91	23.0	5.0	22	503	FALSE	FALSE
15	1-218-289-91	16.0	2.0	26	502	FALSE	FALSE
16	1-216-067-91	19.0	2.0	16	501	FALSE	FALSE
17	8-729-920-86	22.0	3.0	12	504	FALSE	FALSE
18	1-216-809-91	18.0	2.0	24	503	FALSE	FALSE
19	1-216-833-91	10.0	3.0	20	502	FALSE	FALSE
20	1-162-923-91	17.0	6.0	14	501	FALSE	FALSE
21	1-125-891-91	12.0	6.0	10	504	FALSE	FALSE
22	1-216-186-91	12.0	3.0	22	503	FALSE	FALSE
23	1-218-289-91	13.0	6.0	26	502	FALSE	FALSE
24	1-162-923-91	10.0	5.0	14	501	FALSE	FALSE
25	1-469-152-21	20.0	5.0	30	504	FALSE	FALSE
26	1-216-809-91	12.0	5.0	24	503	FALSE	FALSE
27	1-218-289-91	9.0	6.0	26	502	FALSE	FALSE
28	1-216-067-91	11.0	6.0	16	501	FALSE	FALSE
29	1-242-772-91	16.0	4.0	6	504	FALSE	FALSE
30	1-216-809-91	8.0	5.0	24	503	FALSE	FALSE
31	1-218-289-91	9.0	5.0	26	502	FALSE	FALSE
32	1-216-067-91	18.0	3.0	16	501	FALSE	FALSE
33	1-125-891-91	24.0	3.0	10	504	FALSE	FALSE
34	1-216-186-91	13.0	6.0	22	503	FALSE	FALSE
35	1-242-772-91	20.0	4.0	6	000	TRUE	FALSE
36	XXXXXXX	8.0	5.0	18	000	TRUE	FALSE
37	8-729-424-02	22.0	6.0	28	504	FALSE	FALSE
38	8-729-920-75	8.0	3.0	4	000	TRUE	FALSE
39	XXXXXXX	12.0	6.0	18	000	TRUE	FALSE
40	XXXXXXX	12.0	6.0	18	000	TRUE	FALSE
41	8-729-920-75	24.0	4.0	4	504	FALSE	FALSE
42	8-729-920-75	18.0	5.0	4	000	TRUE	FALSE
43	XXXXXXX	24.0	3.0	18	000	TRUE	FALSE
44	XXXXXXX	24.0	3.0	18	000	TRUE	FALSE

Seq	Compo name	X	Y	Feeder	Nozzle	Dummy	Troca	(cont.)
45	1-242-772-91	24.0	5.0	6	504	FALSE	FALSE	
46	8-729-920-86	22.0	4.0	12	000	TRUE	FALSE	
47	XXXXXXXX	22.0	4.0	18	000	TRUE	FALSE	
48	XXXXXXXX	22.0	4.0	18	000	TRUE	FALSE	
49	1-125-891-91	12.0	4.0	10	504	FALSE	FALSE	
50	8-719-058-24	22.0	5.0	8	000	TRUE	FALSE	
51	XXXXXXXX	22.0	6.0	18	000	TRUE	FALSE	
52	XXXXXXXX	22.0	6.0	18	000	TRUE	FALSE	
53	8-729-920-75	18.0	6.0	4	504	FALSE	FALSE	
54	XXXXXXXX	22.0	6.0	18	000	TRUE	FALSE	
55	XXXXXXXX	22.0	6.0	18	000	TRUE	FALSE	
56	XXXXXXXX	22.0	6.0	18	000	TRUE	FALSE	
57	1-242-772-91	20.0	4.0	6	504	FALSE	FALSE	
58	8-719-058-24	22.0	5.0	8	000	TRUE	FALSE	
59	8-729-920-75	8.0	3.0	4	000	TRUE	FALSE	
60	8-729-920-75	18.0	5.0	4	000	TRUE	FALSE	
61	8-729-920-86	22.0	4.0	12	504	FALSE	FALSE	
62	XXXXXXXX	0.0	0.0	0	000	TRUE	FALSE	
63	XXXXXXXX	0.0	0.0	0	000	TRUE	FALSE	
64	XXXXXXXX	0.0	0.0	0	000	TRUE	FALSE	
65	8-719-058-24	22.0	5.0	8	504	FALSE	FALSE	
66	8-729-920-75	8.0	3.0	4	000	TRUE	FALSE	
67	8-729-920-75	18.0	5.0	4	000	TRUE	FALSE	
68	XXXXXXXX	0.0	0.0	0	000	TRUE	FALSE	
69	8-729-920-75	8.0	3.0	4	504	FALSE	FALSE	
70	8-729-920-75	18.0	5.0	4	000	TRUE	FALSE	
71	XXXXXXXX	0.0	0.0	0	000	TRUE	FALSE	
72	XXXXXXXX	0.0	0.0	0	000	TRUE	FALSE	
73	8-729-920-75	18.0	5.0	4	504	FALSE	FALSE	
74	XXXXXXXX	0.0	0.0	0	000	TRUE	FALSE	
75	XXXXXXXX	0.0	0.0	0	000	TRUE	FALSE	
76	XXXXXXXX	0.0	0.0	0	000	TRUE	FALSE	

Tempo de montagem: 18.329 segundos.
Numero de ciclos: 19.

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo