5
Projeto de Novos Polímeros Condutores

Polímeros condutores constituem uma nova classe de materiais eletrônicos com propriedades incomuns, baseadas em novos fenômenos físicos, tendo aplicações com largo potencial tecnológico. A busca por novas estruturas com condutividade elétrica vem sendo feita por pesquisadores em todo o mundo com a finalidade de se criar novos materiais. Uma forma de encontrá-las é explorando o conceito de copolimerização.

Em geral, os copolímeros possuem propriedades elétricas e mecânicas intermediárias de seus relacionados homopolímeros. Devido à rica reatividade do carbono, um número quase infinito de estruturas é possível. Isso faz com que a busca sistemática por novas estruturas seja quase impossível e a abordagem de tentativa e erro vem sendo a regra. Nesta dissertação, uma metodologia sistemática é apresentada, utilizando Algoritmos Genéticos com avaliação distribuída.

5.1.
Introdução

Polímeros são compostos formados por uma repetição mais ou menos regular de um número grande de grupos atômicos (unidades) conectados por ligações químicas formando longas cadeias lineares ou ramificadas, ou redes tri-dimensionais (figuras 75, 76 e 77), cujo processo de formação é chamado de Polimerização. [50][51][52]. De acordo com sua composição, os polímeros podem ser classificados em orgânicos e inorgânicos. As unidades repetitivas que compõem um polímero são equivalentes ou aproximadamente equivalentes ao monômero.

O tamanho da cadeia polimérica é especificado pelo número de unidades que compõem a cadeia, o que é chamado de grau de polimerização. O peso molecular de um polímero é o produto do peso molecular de cada unidade vezes o grau de polimerização [51].

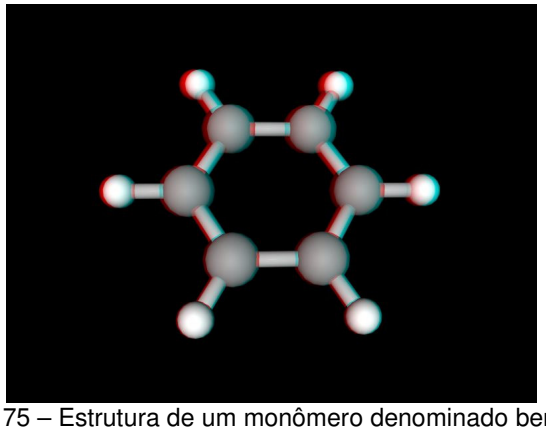

Figura 75 – Estrutura de um monômero denominado benzeno.

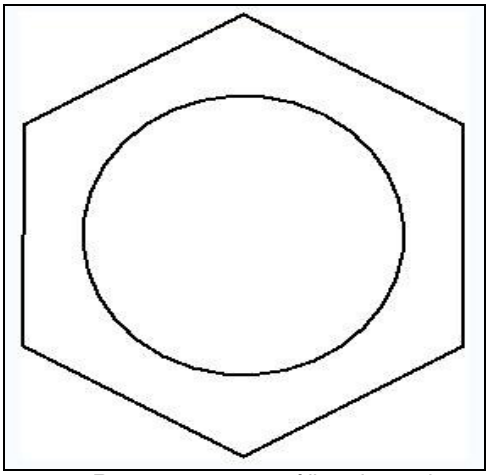

Figura 76 – Representação gráfica de um benzeno.

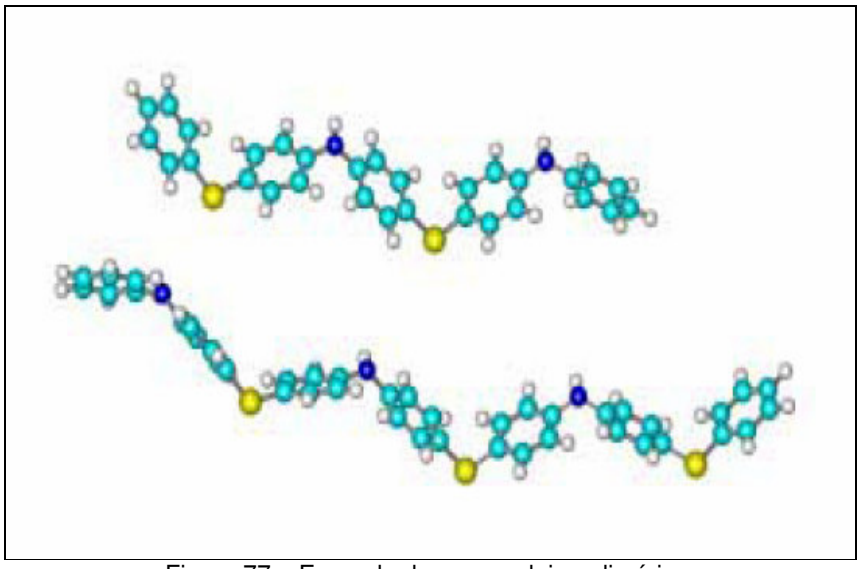

Figura 77 – Exemplo de uma cadeia polimérica.

5.2.
Metodologia

A condutividade elétrica de uma cadeia polimérica está relacionada a sua delocalização eletrônica. Quanto mais delocalizada eletronicamente, maior é a sua condutividade [53]. No cálculo do grau de delocalização, é necessária a descrição da estrutura eletrônica do material.

Para descrever a estrutura eletrônica, foi utilizado o método LCAO (*Linear Combination of Atomic Orbitals*) com o hamiltoniano Hückel (*tight-binding*) [53] que calcula uma matriz hamiltoniana e depois obtém seus autovetores. A partir desse método, utiliza-se o IPN (*Inverse Participation Number*) [53] para medir o grau de delocalização eletrônica, que assume valores entre zero (máxima delocalização) e um (delocalização sobre somente um orbital) [53]. Assim, quanto menor for o IPN, maior é a delocalização.

Portanto, a função de avaliação do AG calcula o IPN de uma cadeia, que é montada através dos genes do cromossomo, e atribui uma nota que é o inverso do IPN pois, neste trabalho, o AG maximiza os valores de *fitness* dos cromossomos, minimizando os valores de IPN. A função de avaliação portanto possui os seguintes passos: montagem da cadeia, cálculo da matriz hamiltoniana, cálculo dos autovetores e cálculo do IPN.

Montagem da cadeia

A construção de uma cadeia depende da quantidade de cada unidade, que é definida por um cromossomo do AG. Neste trabalho, uma unidade é representada por um copolímero – um polímero com dois monômeros. Em [53], o autor utilizou um micro AG com cromossomos binários onde, ao se montar a cadeia fazia-se uma conversão para um número decimal e obtinha-se a quantidade de cada tipo de unidade da cadeia.

Nesse trabalho, foi utilizado um AG convencional, no qual os cromossomos possuem genes com valores reais entre 0 e 1, sendo que a soma dos valores dos genes deveria ser igual a 1, representando 100%. A inicialização de cada cromossomo, assim como os operadores genéticos (*crossover* e mutação), foram implementados levando em consideração tal restrição.

A montagem da cadeia segue alguns passos mostrados na figura 78. De acordo com o número de tipos de unidade que uma cadeia possa conter, o cromossomo do AG é definido. A partir dos valores dos genes do cromossomo e

do tamanho da cadeia, são calculadas as quantidades de cada unidade da cadeia. Em seguida a cadeia é montada com suas respectivas unidades.

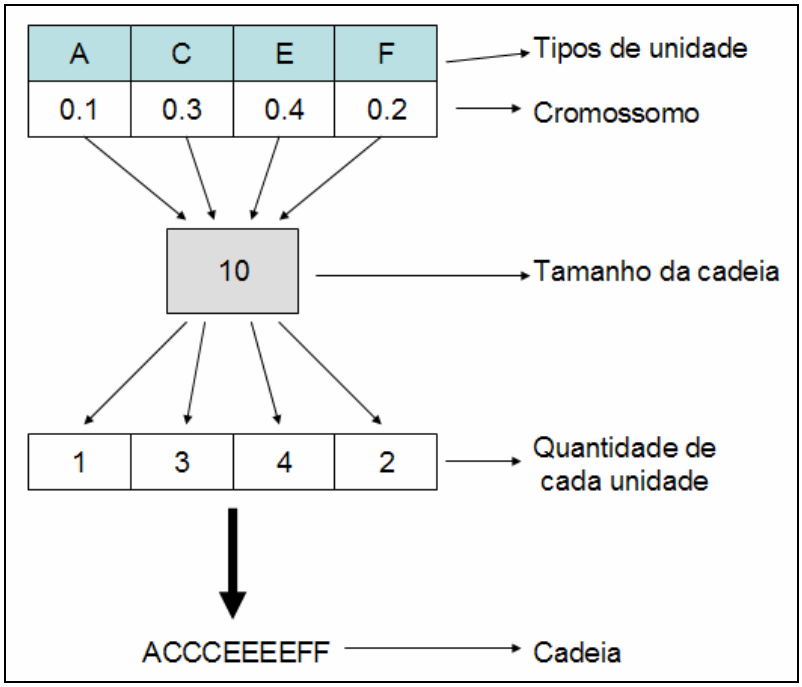

Figura 78 – Passos da montagem de uma cadeia.

Após esses passos, é definida a ordem de cada unidade na cadeia. Neste trabalho, foi utilizada a ordem aleatória, onde cada unidade possui uma posição aleatória na cadeia. Antes da execução do AG, uma única ordem de todas as cadeias a serem geradas é pré-definida e, a partir dessa ordem, as unidades são posicionadas, como pode ser visto na figura 79.

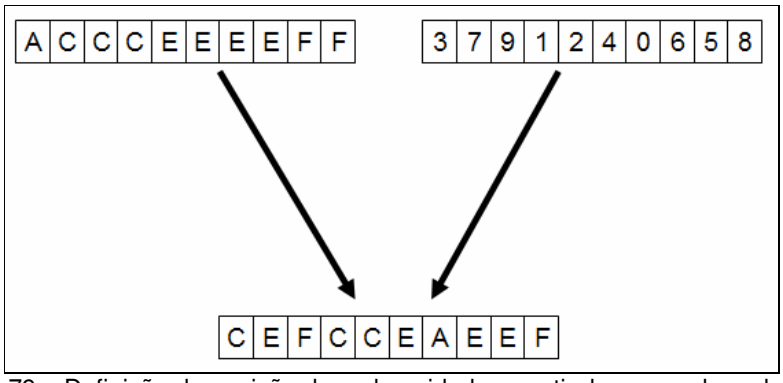

Figura 79 – Definição da posição de cada unidade a partir de uma ordem aleatória.

Cálculo da matriz hamiltoniana

Seguindo as unidades utilizadas em [53] como mostra a figura 80, a tabela 8 apresenta os valores de sítio e ligação das unidades.

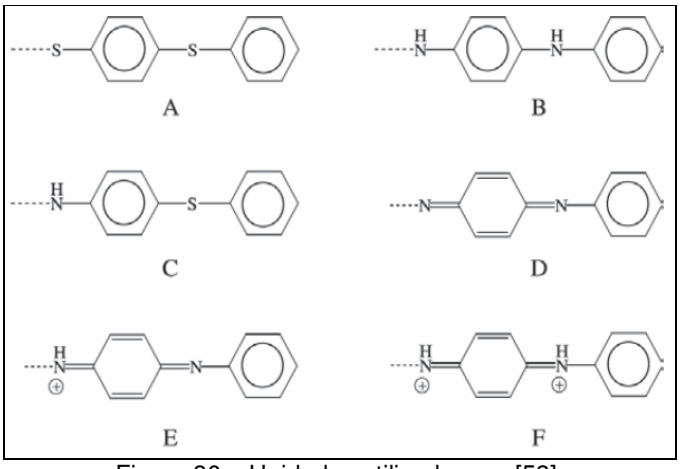

Figura 80 – Unidades utilizadas em [53].

Sítio	α	Ligação	β
C: C-NH	-0,15	C_α - C_β (a)	-0,90
C: C-N=	-0,05	C_α = C_β (a)	-1,10
C: C-NH^+=	-0,20	C_α - C_β (b)	-0,90
C: C-CH_3	0,50	C_α = C_β (b)	-1,00
C: C	0,00	C-NH	-0,80
N: -NH-	-1,50	C-N=	-0,80
N: -N=	-0,50	C-NH^+=	-0,80
N: -NH^+=	-2,00	C=N-	-1,00
		C=NH^+-	-1,00
S: S	-1,30	S-C	-0,60

Tabela 8 – Parâmetros de Hückel das unidades estudadas.

Cada unidade monomérica possui 14 posições, apresentadas na figura 81. A partir dos valores de cada posição, a matriz hamiltoniana é calculada. Assim, uma única unidade é representada por uma matriz de dimensão 14 por 14. Logo, se uma cadeia possuir 100 unidades, sua matriz representante terá dimensão de 1400 por 1400. A figura 82 mostra como é calculada a matriz. Os parâmetros α e β correspondem, respectivamente, ao sítio e à ligação.

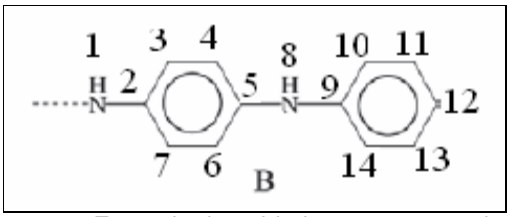

Figura 81 – Exemplo de unidade com suas posições.

$$\begin{vmatrix} (\alpha-\varepsilon) & \beta_{12} & \beta_{13} & \beta_{1n} \\ \beta_{21} & (\alpha-\varepsilon) & \beta_{23} & \beta_{2n} \\ . & & & \\ . & & & \\ \beta_{n1} & \beta_{n2} & \beta_{n3} & (\alpha-\varepsilon) \end{vmatrix}$$

Figura 82 – Cálculo da matriz hamiltoniana.

Cálculo dos autovetores

Em [53], o autor utiliza métodos de aproximação para obter os autovetores da matriz Hamiltoniana, visto que as matrizes são muito grandes. Neste trabalho, os cálculos foram feitos pelo software Matlab utilizando a função “eig”, que retorna uma lista de autovetores de uma determinada matriz de entrada.

Cálculo do IPN

O IPN, que mede o grau de delocalização eletrônica, é calculado através dos valores dos autovetores. Como em [53], neste trabalho, a busca é feita por cadeias que tenham maior delocalização em HOMO (*Highest Occupied Molecular Orbital*). Isso significa que o IPN será calculado através do autovetor relativo a esse orbital. Se a matriz hamiltoniana possui dimensão *n* por *n*, o autovetor relativo a esse orbital é o autovetor *n*/2.

Devido ao cálculo dos autovetores ser computacionalmente muito intenso, consumindo uma grande parte do tempo da avaliação e conseqüentemente da evolução do AG, a distribuição do processamento das avaliações se fez necessária. Neste caso, se uma matriz hamiltoniana tivesse dimensão de 1400 por 1400, como em [53], o processamento da avaliação poderia demorar em

torno de 1 minuto. Se o número de gerações do experimento fosse 50 e a população tivesse 20 indivíduos, o experimento poderia demorar cerca de 5 horas e meia, utilizando um *steady state* inicial de 40% e final de 20%.

Partindo dessa abordagem de distribuição de avaliação da população e reutilizando a maioria dos componentes do GACOM (ver apêndice 1), onde somente a interface Evaluation do módulo do processo de avaliação foi implementada pela classe DistributionManager (figura 83), foi construído um *plug-in* para gerenciar o processo de avaliação distribuída do GACOM (ver apêndice 1).

O *plug-in* foi desenvolvido utilizando o *Microsoft .NET Remoting* [54], um *framework* da Microsoft que permite a criação de aplicações distribuídas utilizando objetos remotos. A criação desses objetos remotos é controlada pelo *framework*, que atende às solicitações de criação dos objetos e os cria, se ainda não existirem. Para que o objeto possa ser criado, ele deve ser registrado, o que também é feito pelo *framework*.

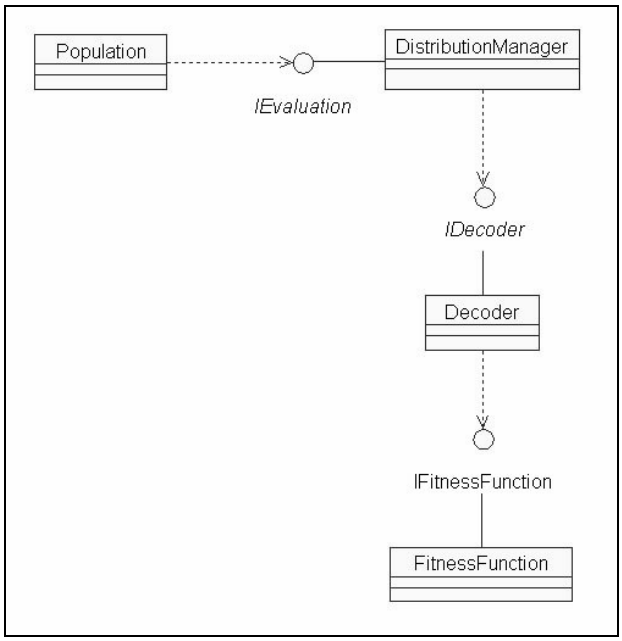

Figura 83 – Modelo de classes da implementação da distribuição da avaliação.

Como esses objetos utilizados não são ativados pelo cliente, é necessário que exista um processo servidor que crie esses objetos e espere por requisições do cliente. Então, o *plug-in* tornou-se uma aplicação distribuída, que utiliza dois processos: o processo cliente e o processo servidor. Esses dois processos trocam informações através de um canal TCP, como mostra a figura 84.

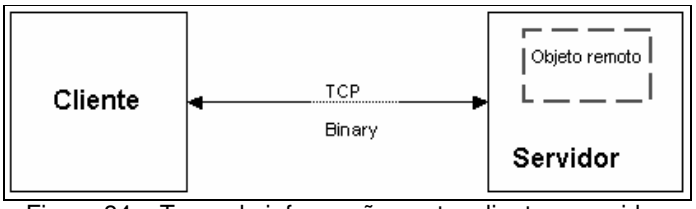

Figura 84 – Troca de informações entre cliente e servidor.

O processo cliente é composto pelos componentes do GACOM, com exceção do componente *Decoder* e do componente *FitnessFunction*, que são componentes do processo servidor. O processo de avaliação normal do GACOM é o seguinte: o componente *Population*, que contém um conjunto de componentes *Individual*, solicita ao componente *Evaluation* sua avaliação. Então, o componente *Evaluation* passa para o *Decoder* essa solicitação e o componente *FitnessFunction* avalia o componente *Individual*.

O processo de avaliação distribuída do *plug-in* é praticamente o mesmo do processo normal. A única diferença é no componente *Evaluation*. No *plug-in*, o componente *Evaluation* foi substituído pelo componente *DistributionManager* (figura 83). Portanto, o componente *DistributionManager* recebe uma solicitação de avaliação do componente *Population*. Então, a população recebida pelo componente *DistributionManager* é dividida em um número de subgrupos pré-especificados no componente e enviadas aos componentes *Decoder* criados, como mostra a figura 85.

Esses componentes então avaliam os subgrupos utilizando um componente *FitnessFunction* específico da aplicação e os retorna ao componente *DistributionManager*, que junta novamente todos os subgrupos em uma população e a retorna ao componente *Population*.

Pode-se perceber, portanto, que somente um componente do GACOM (ver apêndice 1) precisou ser re-escrito. Essa é uma das grandes vantagens da utilização dos componentes de software, a reutilização de código, e uma das principais características do GACOM.

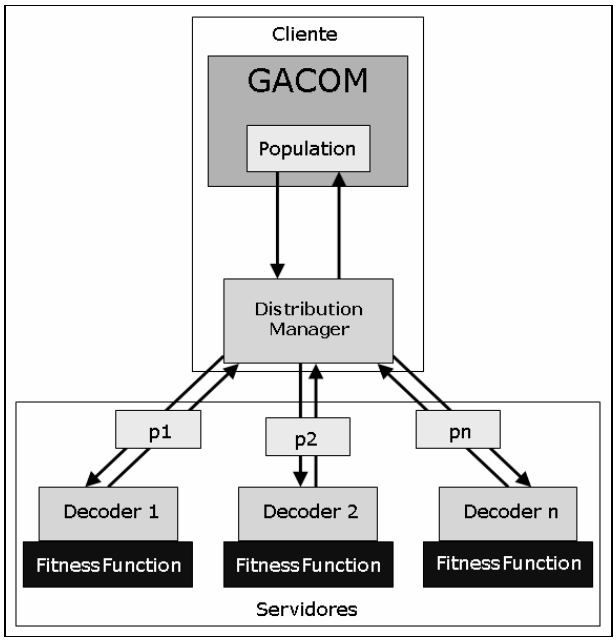

Figura 85 – Processo de distribuição de avaliação do *plug-in*.

5.3. Experimentos

Para comparar os resultados com [53], 4 experimentos foram realizados: dois experimentos com ligas binárias com as unidades A e F, e C e F, respectivamente; um experimento com ligas ternárias com as unidades A, C e F; e um experimento com ligas quintenárias com as unidades A, B, C, D e F. Apesar de [53] apresentar outros experimentos, esses foram escolhidos por considerarem o *gap* igual a zero, como neste trabalho. Além disso, o número de unidades das cadeias (100) também foi mantido. Cadeias com tamanho de 100 unidades permitem uma boa estatística da desordem com baixo custo computacional e também permitem que se obtenha números inteiros de concentrações. A seguir são apresentados os resultados dos 4 experimentos:

Ligas binárias com unidades A e F

- Gerações: 40
- População: 20
- *Crossover*: [0,8 – 0,75]
- Mutação: [0,3 – 0,5]
- *Steady State*: [0,4 – 0,2]

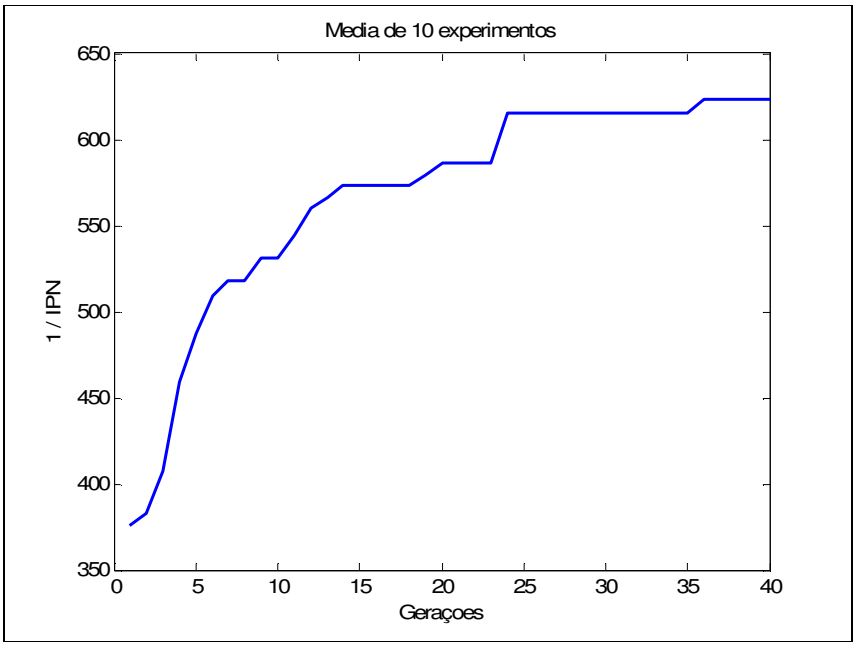

Figura 86 – Curva de evolução dos experimentos com as unidades A e F.

Ligas binárias com unidades C e F

- Gerações: 40
- População: 20
- *Crossover*: [0,8 – 0,75]
- Mutação: [0,3 – 0,5]
- *Steady State*: [0,4 – 0,2]

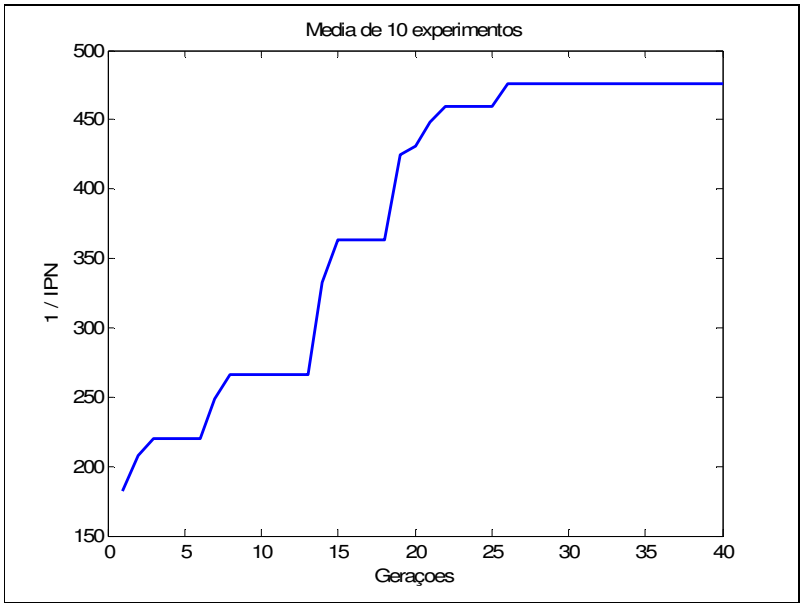

Figura 87 – Curva de evolução dos experimentos com as unidades C e F.

Ligas ternárias com unidades A, C e F

- Gerações: 60
- População: 30
- *Crossover*: [0,8 – 0,75]
- Mutação: [0,3 – 0,5]
- *Steady State*: [0,4 – 0,2]

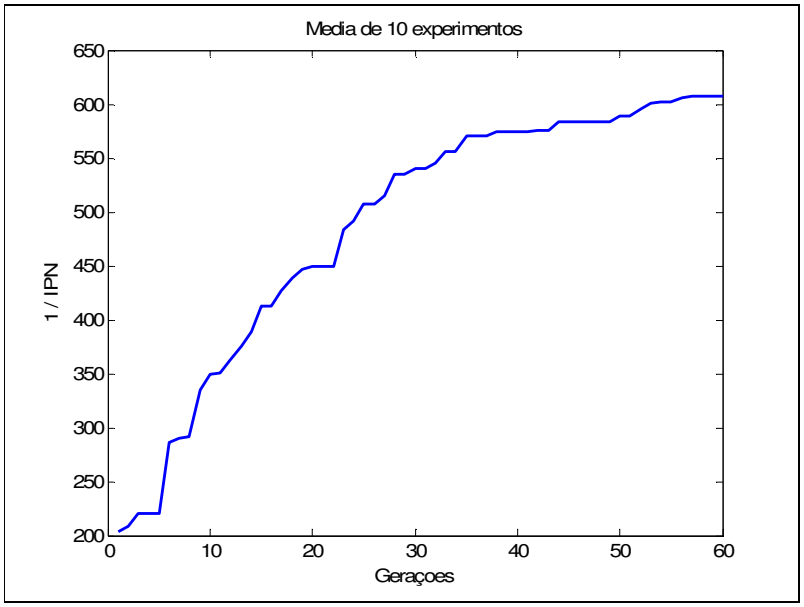

Figura 88 – Curva de evolução dos experimentos com as unidades A, C e F.

Ligas quintenárias com unidades A, B, C, D e F

- Gerações: 100
- População: 50
- *Crossover*: [0,8 – 0,75]
- Mutação: [0,3 – 0,5]
- *Steady State*: [0,4 – 0,2]

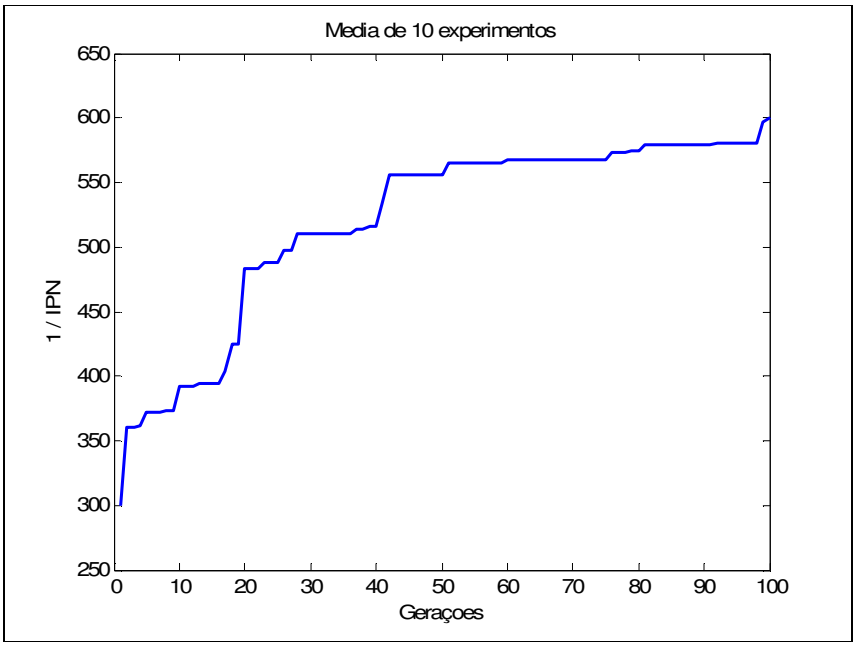

Figura 89 – Curva de evolução dos experimentos com as unidades A, B, C, D e F.

A tabela 9 apresenta os resultados obtidos nos experimentos das figuras 86, 87, 88 e 89, assim como os resultados obtidos por [53] para efeito de comparação. Nos 4 experimentos, os resultados obtidos neste trabalho foram melhores. A utilização de cromossomos com valores reais ao invés de cromossomos binários possibilita um ganho de eficiência na evolução do AG, visto que os cromossomos binários (em [53] o autor utilizou 7 bits para cada quantidade de cada tipo de unidade) podem gerar números entre 101 e 127 inclusive, que não são válidos para valores de porcentagem, neste caso.

Além disso, um número maior de indivíduos por população pode permitir uma busca mais abrangente no espaço de busca do problema, podendo encontrar soluções melhores, que não foram avaliadas em experimentos com populações menores.

Experimento	Resultado obtido (IPN)	Resultdo em [46] (IPN)
Unidades A e F	0,0016 ($A_{100}F_0$)	0,00551 (A_9F_{91})
Unidades C e F	0,0021 (C_0F_{100})	0,00542 (C_5F_{95})
Unidades A, C e F	0,00163 ($A_{99}C_0F_1$)	0,00498 ($A_4C_{52}F_{44}$)
Unidades A, B, C, D e F	0,00166 ($A_{36}B_{53}C_7D_2F_2$)	0,00376 ($A_7B_5C_1D_{30}F_{57}$)

Tabela 9 – Comparação entre os resultados obtidos e de referência.