Capítulo 4

Simulated Annealing Acoplado Síncrono e Assíncrono

Neste capítulo, duas implementações paralelas do *Simulated Annealing Acoplado* são desenvolvidas para explorar a disponibilidade dos núcleos de processamento e melhorar seu desempenho tanto em tempo de execução quanto em qualidade de solução. O CSA foi inicialmente proposto para explorar o potencial paralelo e mitigar o problema de inicialização de parâmetros enquanto aplicado a problemas de otimização global. A troca de informações combinada a uma estratégia para controlar a variância das probabilidades de aceitação resultam em um algoritmo que é mais robusto com relação à inicialização. Por esta razão, este algoritmo tem sido intensamente usado na literatura para ajustar algoritmos de aprendizado de máquina, como nos trabalhos de Rafiee-Taghanaki et al. (2013), Kamari et al. (2013), Hemmati-Sarapardeh et al. (2014), Mehrkanoon et al. (2015), Mahmoodi, Arabloo e Abdi (2014). As definições do *Simulated Annealing Acoplado* e suas implementação síncrona e assíncrona são detalhadas e comparadas neste capítulo.

4.1 Simulated Annealing Acoplado

O *Simulated Annealing Acoplado* (CSA, do inglês *Coupled Simulated Annealing*) é um método estocástico de otimização global capaz de reduzir a sensibilidade dos parâmetros de inicialização enquanto guia o processo de otimização a execuções quase-ótimas (XAVIER-DE-SOUZA et al., 2010). Ele é baseado no *Simulated Annealing* (KIRKPATRICK et al., 1983) e no Minimizadores Locais Acoplados (CLM, do inglês *Coupled Local Minimizers* (SUYKENS; VANDEWALLE; MOOR, 2001). Nos CLMs, múltiplos otimizadores de descida de gradiente acoplado (do inglês *coupled gradient descent optimizers*) são utilizados porque são mais efetivos do que os otimizadores de descida de gradiente com multiplo início (do inglês *multistart gradient descent optimizers*) (SUY-

Ou seja, é como se eu estivesse rodando vários SAs ao mesmo tempo, cada um gerando uma própria solução e avaliando-a (Se aceita ou não), tal como mostrei na sala de aula. Entretanto, só existe apenas uma temperatura de geração e uma temperatura de aceitação, não importa a quantidade de otimizadores SAs. Estas duas variáveis são compartilhadas entre os otimizadores.

KENS; VANDEWALLE; MOOR, 2001; SUYKENS; YALCIN; VANDEWALLE, 2003).

O CSA consiste em um conjunto de otimizadores SA cujo comportamento individual é similar à execução do *Simulated Annealing*, ou seja, os passos dos algoritmo que envolvem a *geração* e *avaliação* de uma única solução são executados individualmente por cada otimizador. Durante a geração de soluções, a temperatura de geração T_k^{gen} é responsável pelo grau de similaridade entre a solução corrente e a nova geração na iteração k. Assim, quanto maior o valor de T_k^{gen}, maior será a diferença média entre a nova solução e a corrente. A temperatura de aceitação T_k^{ac} utilizada durante a avaliação das nova soluções é responsável pela chance de aceitação de tais soluções, isto é, elevados valores resultam em maiores chances de uma nova solução ser aceita. Existe somente uma única temperatura de geração e de aceitação, independentemente no número de otimizadores.

Diferente do SA, na qual a probabilidade de aceitação $A(\mathbf{x} \to \mathbf{y})$ é em função somente da solução corrente $\mathbf{x}$ e da solução gerada $\mathbf{y}$ (Equação 2.11), a probabilidade de aceitação do CSA é uma função escalar de acordo com

$$0 \leq A_{\Theta}(\gamma, \mathbf{x}_i \to \mathbf{y}_i) \leq 1, \tag{4.1}$$

"m" é o número de otimizadores (ou de soluções correntes)

para todo $\mathbf{x}_i \in \Theta$, $\mathbf{y}_i \in \Omega$, com $\Theta, \Omega, \in \mathbb{R}^n$, sendo Θ o conjunto das soluções correntes e Ω o conjunto de todas as soluções de dimensionalidade n, para $i = 1, \ldots, m$, com o número de elementos m de Θ. O termo γ é responsável pelo acoplamento entre os otimizadores SA, uma vez que ele é uma função da energia de todas as soluções correntes do CSA. A função $N(\mathbf{x}_i, T_k^{\text{gen}})$ gera uma nova solução $\mathbf{y}_i$ que é alcançada em uma única iteração do algoritmo para a temperatura de geração e solução corrente $\mathbf{x}_i$ de acordo com uma distribuição escolhida. Assim como no SA, $N(\mathbf{x}_i, T_k^{\text{gen}})$ pode ser utilizada a partir da distribuição de Cauchy, de acordo com:

A fórmula para gerar esta equação é a mesma do SA e está descrita no DOCUMENTO que descreve o trabalho.

$$\mathbf{y}_i = \mathbf{x}_i + \varepsilon * T_k^{\text{gen}}, \tag{4.2}$$

onde cada elemento de $\varepsilon \in \mathbb{R}^n$ é uma variável independente aleatória amostrada da distribuição de Cauchy. O escalonamento de resfriamento também pode ser linear. Ele é descrito da seguinte forma:

A fórmula de escalonamento da tempertura de geração está descrita no DOCUMENTO que descreve o trabalho.

$$T_k^{\text{gen}} = \frac{T_1^{\text{gen}}}{k+1}, \tag{4.3}$$

sendo T_1^{gen} a temperatura de geração inicial.

A probabilidade de aceitação do CSA pode assumir diferentes formas, tal como o SA. A probabilidade de aceitação inicialmente descrita por Xavier-de-Souza et al. (2010) é

Não utilizar estas equações!

$$A_\Theta(\gamma, \mathbf{x}_i \to \mathbf{y}_i) = \frac{\exp\left(\frac{E(\mathbf{x}_i)}{T_k^{\text{ac}}}\right)}{\gamma} \tag{4.4}$$

$$\gamma = \sum_{\forall \mathbf{x}_j \in \Theta} \exp\left(\frac{E(\mathbf{x}_j)}{T_k^{\text{ac}}}\right). \tag{4.5}$$

Observe que as energias não são limitadas numericamente. Caso elas sejam positivas, totalmente ou parcialmente, poderá surgir instabilidade numérica na avaliação da função de probabilidade de aceitação. Para muitas funções, este problema pode ser facilmente resolvido através da normalização da função objetivo de tal modo que $\forall \mathbf{x} \in \Omega, E(\mathbf{x}) \leq 0$. Entretanto, com funções cujo limites são desconhecidos, é necessário utilizar outra abordagem. Assim, Xavier-de-Souza et al. (2010) sugerem que todas as energias em (4.4) and (4.5) sejam subtraídas da energia máxima corrente. A equação resultante, descrita a seguir, é numericamente mais atrativa porque todos os expoentes são negativos.

Utilizar estas equações!

$$A^*_\Theta(\gamma^*, \mathbf{x}_i \to \mathbf{y}_i) = \frac{\exp\left(\frac{E(\mathbf{x}_i) - max(E(\mathbf{x}_i))_{\mathbf{x}_i \in \Theta}}{T_k^{\text{ac}}}\right)}{\gamma^*}, \tag{4.6}$$

$$\gamma^* = \sum_{\forall \mathbf{x} \in \Theta} \exp\left(\frac{E(\mathbf{x}) - max\left(E\left(\mathbf{x}_i\right)\right)_{\mathbf{x}_i \in \Theta}}{T_k^{\text{ac}}}\right), \tag{4.7}$$

Tal como foi definida, a função de probabilidade de aceitação A^*_Θ tem como característica

Meramente formal. Não iremos utilizar no código.

$$\sum_{\forall \mathbf{x}_i \in \Theta} A^*_\Theta(\gamma,\ \mathbf{x}_i \to \mathbf{y}_i) \equiv \sum_{\forall \mathbf{x}_i \in \Theta} A^*_\Theta = 1. \tag{4.8}$$

Além disso, é fácil notar a partir de (4.6) que em altas temperaturas, independentemente da energia das soluções correntes, todas as probabilidades de aceitação se aproximam de $1/m$, enquanto que para uma temperatura suficientemente baixa, a probabilidade da maior energia se aproxima de 1 e todas as outras se aproximam de 0. Estes dois casos correspondem aos dois limites para a variância e podem ser escritos como:

$$\frac{1}{m} \leq \sum_{\forall \mathbf{x}_i \in \Theta} A^2_\Theta \leq 1. \tag{4.9}$$

Os benefícios que o acoplamento pode oferecer vão além de uma pura troca de informação e podem ser utilizados para controlar medidas gerais estatísticas, possibilitando modificar alguns parâmetros de controle que pode ter influência crucial na otimização.

Xavier-de-Souza et al. (2010) utilizaram a temperatura de aceitação T_k^{ac} para controlar a variância da função de probabilidades de aceitação σ^2. A sua definição e seus limites estão descritos a seguir.

Conhecendo (4.8), a variância de A^*_Θ assume a forma

Meramente formal. Não iremos utilizar no código.

$$\begin{aligned}\sigma^2 &= \frac{1}{m}\sum_{\forall \mathbf{x}_i \in \Theta}(A^*_\Theta)^2 - \left(\frac{1}{m}\sum_{\forall \mathbf{x}_i \in \Theta} A^*_\Theta\right)^2, \\ &= \frac{1}{m}\sum_{\forall \mathbf{x}_i \in \Theta}(A^*_\Theta)^2 - \frac{1}{m^2}\ . \end{aligned} \tag{4.10}$$

Aplicando os limites de (4.9) em (4.10), conclui-se que

$$0 \leq \sigma^2 \leq \frac{m-1}{m^2}\ . \tag{4.11}$$

A variância da função de probabilidades de aceitação desempenha um papel significativo na otimização. Um bom valor da variância equilibra a correta relação entre a exploração do espaço de busca e o refinamento das soluções correntes através da busca local. Xavier-de-Souza et al. (2010) mostraram em seus experimentos que o desempenho da otimização do CSA melhora as execuções quando a variância é próxima de 99% de seu valor máximo. Para esta conclusão, eles utilizaram 14 funções objetivos, sendo 2 compostas por funções unimodais, 6 funções multimodais e 6 funções resultantes da rotação das 6 últimas funções listadas. As funções estão descritas no Apendice B. O custo médio foi normalizado sobre 50 execuções e dado em função do percentual da variância desejada. A Figura 4.1 mostra o resultado obtido destes experimentos.

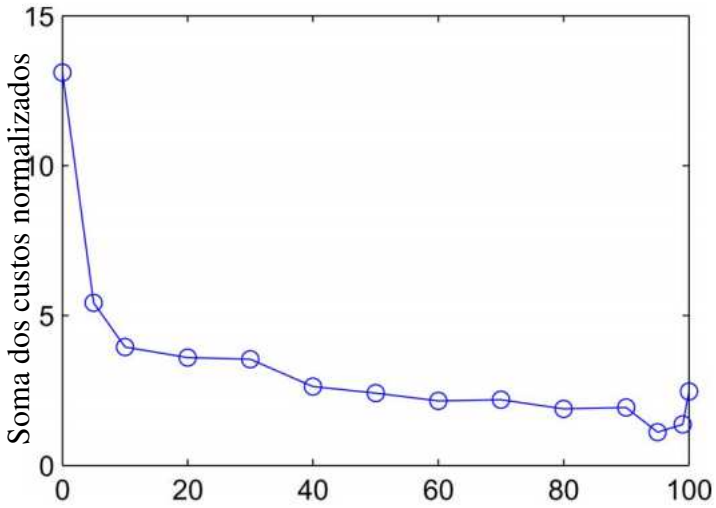
Um simples controle da variância pode, então, ser formulado:

$$\begin{aligned}&\text{se } \sigma^2 < \sigma_D^2, \quad T^{ac} \leftarrow T^{ac}(1-\alpha); \\ &\text{se } \sigma^2 \geq \sigma_D^2, \quad T^{ac} \leftarrow T^{ac}(1+\alpha).\end{aligned} \tag{4.12}$$

onde σ_D^2 é a variância desejada (99% do valor máximo da variância) e α é a taxa de crescimento ou decaimento da temperatura de aceitação, normalmente na faixa de $(0\ ,\ 0,1]$ (XAVIER-DE-SOUZA et al., 2010). Quando o valor da variância está abaixo do valor desejado, a temperatura de aceitação diminui por um fator de $1-\alpha$; caso contrário, o seu valor é multiplicado por $1+\alpha$. Elevados valores de α torna o processo de busca do algoritmo totalmente aleatória.

Utilize 0.01

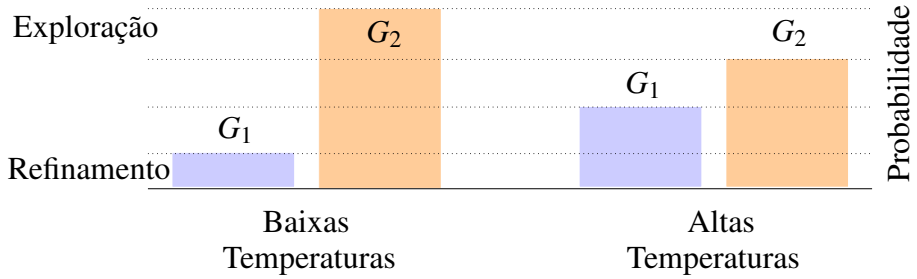
Figura 4.1: Soma dos custos médios normalizados pelo valor da variância desejada σ_D^2 expressa no percentual de seu valor máximo.



Fonte: Xavier-de-Souza (2010).

Quando as soluções possuem custo energético parecidos, a variância é menor e, portanto, o controle tende a reduzir a temperatura de aceitação na esperança de aumentar a variância. A redução desta temperatura induz os otimizadores com baixas probabilidades de aceitação a realizarem refinamento - reduzirem suas probabilidades de aceitação - enquanto que os otimizadores com maiores probabilidades realizam exploração - aumentam suas probabilidades de aceitação. Quando os custos energéticos das soluções correntes são suficiente diferentes, a variância é maior. Desse modo, a temperatura de aceitação será aumentada na tentativa de reduzir a variância. O aumento da temperatura induz os otimizadores a terem as probabilidades de aceitação parecidas. Portanto, aqueles com menores probabilidades de aceitação terão suas chances aumentada, enquanto que os otimizadores com maiores probabilidades de aceitação terão tais chances reduzidas. Esta qualidade, que caracteriza o CSA como um processo que tenta evitar a estagnação do procedimento de busca, é descrita na Figura 4.2.

Figura 4.2: Comportamento da probabilidade de aceitação dos otimizadores SA com a variação da temperatura. $G1$ representa os otimizadores com menores probabilidades de aceitação e $G2$ aqueles com as altas probabilidades. Caso a temperatura aumente, há maiores chances das probabilidades de aceitação de $G1$ e $G2$ se aproximarem. Caso haja redução da temperatura, tais probabilidades terão maiores chances de se distanciarem.



Fonte: Gonçalves-e-Silva (2013).

Com o controle da variância, não é necessário especificar o escalonamento da temperatura de aceitação nem realizar o exaustivo processo empírico de sua valoração inicial. Em contrapartida, duas novas variáveis são introduzidas α e σ_D^2, mas eles possuem uma faixa de operação bem definida e são menos dependentes do problema de otimização.

O algoritmo da implementação do CSA está definido no Algoritmo 11.

Algoritmo 11: Pseudocódigo do *Coupled Simulated Annealing*

Passo 1) Inicialização:

- Atribua m soluções iniciais aleatórias a Θ;
- Avalie as energias $E(\mathbf{x}_i)$, $\forall \mathbf{x}_i \in \Theta$;
- Defina o número da iteração $k = 1$;
- Defina as temperaturas iniciais T_k^{gen} and T_k^{ac};
- Atribua um valor a α;
- Avalie o termo de acoplamento γ;
- Defina $\sigma_D^2 = 0,99\frac{(m-1)}{m^2}$.

Passo 2) Geração:

- Gere uma nova solução $\mathbf{y}_i = \mathbf{x}_i + \varepsilon * T_k^{\text{gen}}$, $\forall i = 1, \cdots, m$, de acordo com (4.2) ;
- Avalie as energias $E(\mathbf{y}_i), \forall i = 1, \ \ldots \ , m$.

Passo 3) Avaliação:

- $\forall i = 1, \cdots, m$, faça $\mathbf{x}_i \leftarrow \mathbf{y}_i$ se:

A) $E(\mathbf{y}_i) \leq E(\mathbf{x}_i)$; ou

B) $A_\Theta^* > r$, onde r é um número aleatório amostrado da distribuição uniforme $[0, 1]$.

Passo 4) Atualização:

- Calcule σ^2;
- Ajuste T_k^{ac} de acordo com a regra a seguir:
  - Se $\sigma^2 < \sigma_D^2$ então $T_{k+1}^{\text{ac}} = T_k^{\text{ac}}(1 - \alpha)$;
  - Se $\sigma^2 > \sigma_D^2$ então $T_{k+1}^{\text{ac}} = T_k^{\text{ac}}(1 + \alpha)$.
- Reduza a temperatura de geração T_k^{gen} de acordo com o escalonamento escolhido;
- Avalie o termo de acoplamento γ;
- Incremente k em 1.

Passo 5) Critério de Parada:

- Pare se o critério de parada foi alcançado;
- Senão volte ao Passo 2.

4.2 Implementações Paralelas do CSA

As metaheurísticas permitem uma redução significativa do tempo para encontrar uma solução razoável. No entanto, muitas das abordagens originais são completamente sequenciais e não aproveitam ao máximo as novas arquiteturas paralelas atuais. Por essa razão, várias metaheurísticas foram reformuladas para explorar seu potencial paralelo, reduzindo o tempo de processamento e também melhorando a qualidade das soluções forne-

cidas. Algumas destas formulações foram direcionadas para problemas específicos, como os algoritmos de Lou e Reinitz (2016), Liu e Wang (2015), Delévacq et al. (2013), Subramanian et al. (2010), Hong e He (2011), Santander-Jiménez e Vega-Rodríguez (2015).

Outras metaheurísticas paralelas são mais gerais e podem ser aplicadas a diferentes tipos de problemas. Isso inclui a análise de impacto da topologia de migração no desempenho de algoritmos de otimização global paralela que usam modelo de ilha (RUCIŃSKI; IZZO; BISCANI, 2010); uma nova diretriz para o desenho e implementação de metaheurísticas efetivas de busca local em GPU (LUONG; MELAB; TALBI, 2013a); três metaheurísticas híbridas (YI; DUAN; LIAO, 2013) que melhoram os dois algoritmos metaheurísticos de evolução diferencial híbridos descritos por (LIAO, 2010); um novo algoritmo de nuvem paralelo para problemas de alocação de redundância de sistemas de sensores inteligentes na Internet das Coisas (YEH; LIN, 2016); e um novo método de otimização global paralelo assíncrono baseado no SA e DE (OLENŠEK et al., 2011).

Recentemente, Unidades de Processamento Gráfico (GPUs) surgiram como uma alternativa para acelerar o tempo computacional em dezenas de vezes, incluindo metaheurísticas (SOUZA; SANTOS; COELHO, 2017). O desenvolvimento de heurísticas baseadas em GPU podem ser encontradas em nos trabalhos de Ament et al. (2010), Kider et al. (2010), Liu et al. (2012), Ding, Zheng e Tan (2013), Luong, Melab e Talbi (2013b), Wilton et al. (2015), Zhou e Zeng (2015). Algumas estratégias heurísticas híbridas CPU-GPU podem ser encontrado nos trabalhos desenvolvidos por Kadjo et al. (2015), Iturriaga et al. (2015), Coelho et al. (2016), Souza, Santos e Coelho (2017), Vidal, Alba e Luna (2017).

O CSA original, desenvolvido por Xavier-de-Souza (2007), foi implementado de forma síncrona em uma arquitetura de memória distribuída utilizando MPI (XAVIER-DE-SOUZA, 2007; MPI, 2018). Neste trabalho, foi desenvolvido duas implementações paralelas do CSA para arquitetura de memória compartilhada, o SCSA (*Simulated Annealing Acoplado Síncrono*, do inglês *Synchronous Coupled Simulated Annealing*) e o ACSA (*Simulated Annealing Acoplado Assíncrono*, do inglês *Asynchronous Coupled Simulated Annealing*). Embora seja bem conhecido que as arquiteturas de memória compartilhada não são muito adequadas para um número muito grande de processadores, os nós de computação de sistemas de computação de alto desempenho geralmente implementam um espaço de endereço compartilhado (VAJDA, 2011). Não está claro em que ponto no futuro a indústria fará o movimento em direção aos chips de memória distribuída, o que justifica as implementações descritas e analisadas. Os códigos-fonte estão disponibilizados em https://gitlab.com/groups/lappsufrn/CSA.

4.2.1 Simulated Annealing Acoplado Síncrono

O Algoritmo 12 e a Figura 4.3 mostram a implementação paralela do *Simulated Annealing Acoplado Síncrono* (SCSA). O estágio *Inicialização* é executado sequencialmente, define as temperaturas iniciais, dimensão e calcula a variância desejado. Finalmente, a região paralela é criada com o número de otimizadores m selecionado pelo usuário. Cada otimizador corresponde a uma *thread*.

Muito importante. Vai facilitar a implementação

No estágio *Inicialização Paralela*, cada otimizador i cria uma solução aleatória $\mathbf{x}_i$ e avalia seu custo. O estágio termina com uma barreira paralela, ou seja, o estágio *Avaliação de Gamma* só pode ser executado após o término do estágio *Inicialização Paralela*.

O termo de acoplamento γ precisa ser avaliado antes que o laço principal seja iniciado. Uma avaliação incorreta deste termo pode resultar no cálculo errôneo da função de probabilidade de aceitação A_Θ no estágio *Aceitação*. Apenas um otimizador avalia γ enquanto os outros esperam bloqueados na barreira antes de *Geração*.

O laço principal corresponde aos estágios de *Geração* a *Critério de Parada*. Cada otimizador gera e calcula a energia de suas soluções $\mathbf{y}_i$ geradas a partir de $\mathbf{x}_i$. No estágio *Aceitação*, cada otimizador avalia sua solução com base na função de probabilidade de aceitação A_Θ. Antes do estágio *Atualização*, há uma barreira, ou seja, o estágio *Atualizaão* só é executado quando todos os otimizadores no estágio *Avaliação* concluirem suas atividades. Isso ocorre porque é necessário fazer com que todas as soluções atuais calculem σ_D^2 e atualizem as temperaturas. Apenas um otimizador executa o estágio *Atualização*. O algoritmo para quando o *Critério de Parada* é atendido, caso contrário, ele retorna ao quarto estágio.

Algoritmo 12: Algoritmo *Simulated Annealing Acoplado Síncrono*

1) Inicialização:

Defina as temperatures T_k^{gen} e T_k^{ac};

Defina a dimensão inicial n;

Defina o número de otimizadores m;

Defina a variância desejada $\sigma_D^2 = 0,99\frac{m-1}{m^2}$;

Crie uma região paralela com m otimizadores.

2) Inicialização Paralela:

Cada otimizador:

a) Atribui sua solução inicial aleatória $\mathbf{x}_i$;

b) Atribui a energia $E(\mathbf{x}_i)$;

Execute uma barreira paralela.

3) Avaliação de Gamma:

Somente um otimizador avalia o termo de acoplamento γ;

Execute uma barreira paralela.

4) Geração:

Cada otimizador gera uma nova solução $\mathbf{y}_i$;

Cada otimizador avalia a energia de sua solução gerada.

5) Aceitação:

Cada otimizador faz $\mathbf{x}_i \leftarrow \mathbf{y}_i$ se:

a) $f(\mathbf{y}_i) \leq f(\mathbf{x}_i)$, ou

b) $A_\Theta > r$, onde r é uma variável aleatória amostrada da distribuição uniforme $[0,1]$.

Execute uma barreira paralela.

6) Atualização:

Somente um otimizador:

a) Ajuste T_k^{ac} de acordo com a seguinte regra:

Se $\sigma^2 < \sigma_D^2$ então $T_{k+1}^{\text{ac}} = T_k^{\text{ac}}(1-\alpha)$,

Se $\sigma^2 > \sigma_D^2$ então $T_{k+1}^{\text{ac}} = T_k^{\text{ac}}(1+\alpha)$;

b) Decremente a temperatura de geração T_k^{gen} de acordo com o escalonamento escolhido;

c) Avalie o termo de acoplamento γ.

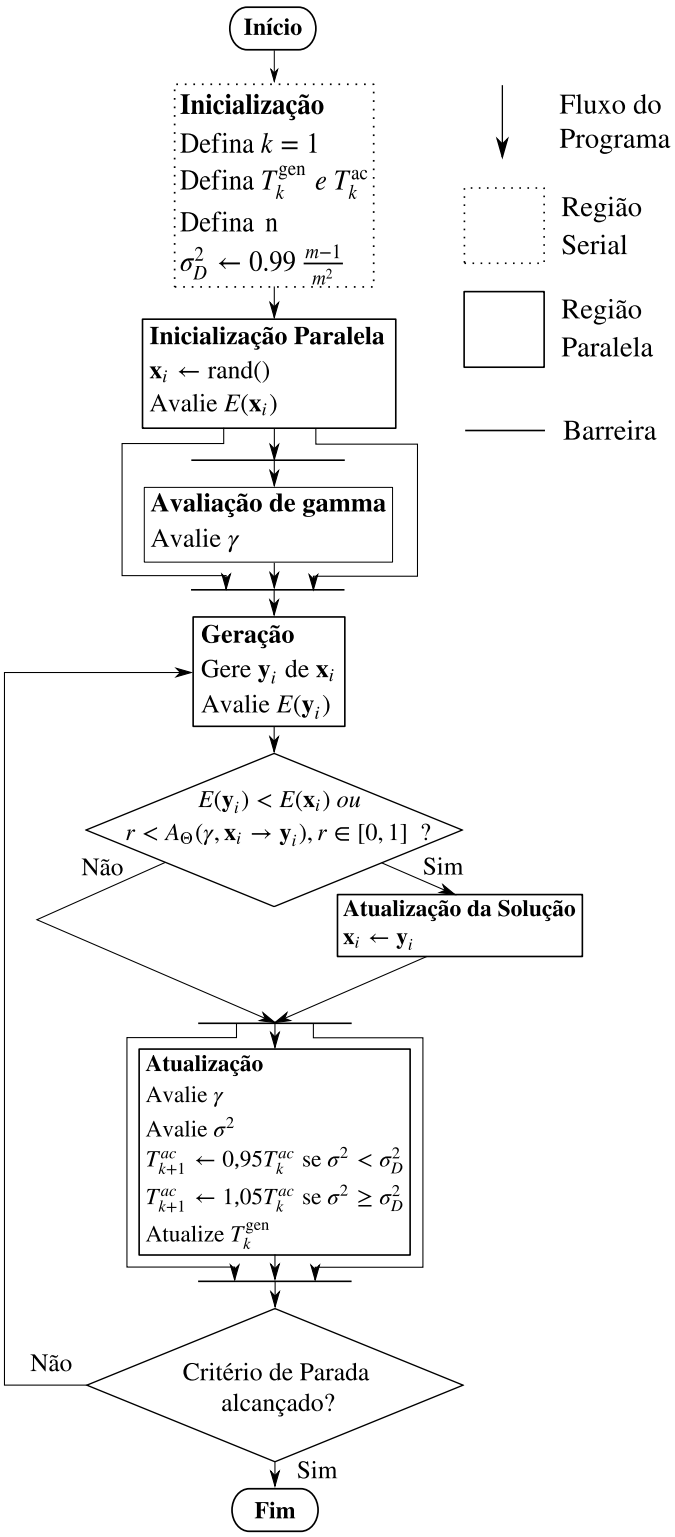
Execute uma barreira paralela.

7) Critério de Parada:

Pare de o critério de parada for alcançado;

Caso contrário, siga ao estágio 4.

Figura 4.3: Implementação paralela síncrona do *Simulated Annealing Acoplado*. Diversas sincronizações são necessárias para o cálculo consistente de A_Θ.



Fonte: Elaboração própria.