Inteligência Computacional - 2

O Amostrador de Gibbs

$\rightarrow$ Método alternativo de amostragem da distribuição de Gibbs

A partir de uma solução arbitrária, novas soluções são geradas como amostras de uma distribuição condicional

Seja $X = \{X_s; s = 1, 2, ..., N\}$ uma solução, com $X_s \in \{x^{(i)}; i = 1, 2, ..., M\}$

Para gerar novas soluções, escolhe-se uma seqüência de visitação para as variáveis X_s, e toma-se, para cada uma delas, uma amostra de

$$p_i \equiv P(X_s = x_i | X_t; t \neq s) = \frac{\pi(X_s = x_i, X_t; t \neq s)}{\Sigma_i \, \pi(X_s = x_i, X_t; t \neq s)}$$

onde $\pi(\cdot)$ é a distribuição de Gibbs

Para obter a amostra de p_i, constróem-se as somas

$$P_j = \sum_{i=1}^{j} p_i = \sum_{i=1}^{j} P(X_s = x_i | X_t; t \neq s)$$

para j=1,2,...,M, gera-se um número aleatório $q \in [0, 1]$, e escolhe-se $X_s = x^{(i)}$, se $P_{i-1} < q < P_i$

Variações do *Annealing*

i) *Annealing Simulado Restrito*

No *annealing* padrão, soluções implausíveis são *inibidas* por termos de *penalização*

Alternativamente, pode-se *proibir* tais soluções, com o uso de termos de *restrição*

$\rightarrow$ Para cada T, o algoritmo gera amostras de uma distribuição de Gibbs modificada,

$$P(X) = \delta_{V_R(X)} \frac{\exp\{-E(X)/T\}}{Z}$$

onde

$$\delta_{V_R(X)} = \begin{cases} 1 & : \quad V_R(X) = 0 \\ 0 & : \quad V_R(X) \neq 0 \end{cases}$$

$\rightarrow$ O *annealing* restrito gera soluções X que minimizam $E(X)$, ao mesmo tempo satisfazendo a restrição $V_R(X) = 0$

Implementação:

O termo de restrição deve ser introduzido com cuidado, para não comprometer a convergência do algoritmo

$\rightarrow$ No *annealing* restrito, emprega-se o algoritmo de Metropolis para gerar amostras de $P(X) = \exp\{-E_t(X)\}/Z$, onde

$$E_t(X) = \frac{E(X) + \lambda(t)V_R(X)}{T(t)}$$

com $T(t) \rightarrow 0$ e $\lambda(t) \rightarrow \infty$

A convergência do algoritmo está assegurada se

$$\frac{\lambda(t)}{T(t)} \leq const. \log t$$

- Exemplo: Aplicação de Visão Computacional

Reconstrução de forma 3D através de *Shape-from-Shading*

$\rightarrow$ Obter a superfície que satisfaz a *equação de irradiância da imagem*,

$$I(s) = R(\hat{n}(s))$$

onde $s = (x, y)$ denota as posições no plano-imagem, $I(s)$ é a imagem observada, e $\hat{n}(s)$ é o mapa de vetores unitários normais à superfície (representação de forma)

Formulação como um problema de otimização restrita:

$\rightarrow$ Minimizar a função custo

$$E(\hat{n}) = -\sum_{<s,s'>} \hat{n}(s).\hat{n}(s') + \lambda \sum_{s} [I(s) - R(\hat{n}(s))]^2$$

onde $< s, s' >$ indica posições vizinhas no plano da imagem

$\rightarrow$ Solução via *annealing* simulado restrito

ii) Aceitação Limiar (*Threshold Accepting*)

$\rightarrow$ A regra de aceitação do *annealing* é mudada para

Se $\Delta E < L$, aceite a nova solução

onde L é um valor limiar escolhido numa seqüência decrescente

$\rightarrow$ Aceitam-se todas as soluções que não são *muito piores* que a solução corrente

• Vantagem: Implementação mais simples e rápida

e.g., Não requer números aleatórios de qualidade, ou o cômputo da exponencial

iii) *Annealing* Microcanônico

$\rightarrow$ Baseado no *ensemble* microcanônico: soluções são geradas a *energia* constante

$\rightarrow$ Emprega o *algoritmo de Creutz*, para gerar amostras da distribuição de equilíbrio de um sistema isolado (i.e., a energia constante)

$$P_\mu(X) = \begin{cases} K & : \quad E_0 - \delta E \leq E(X) \leq E_0 + \delta E \\ 0 & : \quad \text{em outro caso} \end{cases}$$

onde $\delta E << E(X)$ representa uma flutuação de energia

- Algoritmo de Creutz

→ Introduz uma variável (*demônio*) para modelar as flutuações de energia

A partir de uma solução inicial arbitrária, geram-se novas soluções de forma que $E + D = constante$

onde D é a energia do demônio, inicializada com um valor alto

→ Regra de Aceitação:

Se $\Delta E \leq 0$, aceite a nova solução

Se $\Delta E > 0$, aceite a nova solução, desde que $\Delta E < D$

Em ambos os casos, se houve aceitação, faça $D = D - \Delta E$

Annealing Microcanônico

→ Implementação iterativa do algoritmo de Creutz, com o valor inicial do demônio escolhido numa seqüência decrescente

→ O *annealing* é controlado pelo demônio

Proposta Original:

Implementação paralela (1 demônio por variável do problema)

→ O valor de D não cresce muito

Em implementações seqüenciais, é preciso limitar o demônio

• Vantagens do *Annealing* Microcanônico

Implementação mais simples e rápida

Permite estabelecer, com precisão, se o equilíbrio (térmico) foi atingido:

Da física estatística,

Valor médio do demônio é igual à temperatura ($\overline{D} = T$)

→ No equilíbrio térmico, $\overline{D}/\sigma(D) = 1$, onde $\sigma(D)$ é o desvio padrão dos valores do demônio

iv) Otimização Microcanônica

Alternativa mais eficiente ao *annealing* microcanônico

→ Não envolve *annealing*

→ Duas fases que se alternam: *Inicialização* e *Amostragem*

• Inicialização:

→ Melhoramento iterativo: só se aceita uma nova solução se $\Delta E < 0$

→ Leva a um mínimo local

• Amostragem:

$\rightarrow$ Algoritmo de Creutz, com o demônio definido por sua energia inicial, D_I, e sua energia máxima, D_{Max}

$\rightarrow$ Permite que o algoritmo liberte-se do mínimo local, movendo-se para uma nova solução de aproximadamente mesmo custo

Os valores D_I e D_{Max} podem ser obtidos na fase de inicialização:

$\rightarrow$ Compila-se uma lista com os valores de $\Delta E > 0$ propostos naquela fase, e escolhe-se D_I e D_{Max} como duas entradas naquela lista

$\rightarrow$ O demônio reflete o relevo da função custo na região do espaço de soluções pesquisada na inicialização

v) *Annealing* Simulado Rápido

$\rightarrow$ Modificação na amostragem do espaço de soluções

Ao invés de uma amostragem uniforme, emprega-se uma outra que possibilita saltos maiores no espaço de soluções

$\rightarrow$ Baseada na distribuição de Cauchy-Lorentz

$\rightarrow$ Varredura mais eficiente do espaço de soluções permite taxas de resfriamento mais rápidas

vi) *Annealing* de Tsallis

$\rightarrow$ Baseado na reformulação da Mecânica Estatística proposta por C. Tsallis

$\rightarrow$ Probabilidade de Aceitação:

$$P = \min(1, [1 - [1 - (1-q)\Delta E/T]^{1/1-q})$$

com q inteiro

Para $q = 1$: *Annealing* Simulado Tradicional

Para $q < 1$: Mais rápido que a do *annealing* simulado rápido

• Aplicação: Alocação de Tarefas em Sistemas Distribuídos

→ Definir um compartilhamento de recursos computacionais que satisfaça as restrições lógicas e temporais das tarefas

→ Dois passos:

i) Escalonamento Global: Alocação de tarefas aos nós

ii) Escalonamento Local: Seqüenciamento das tarefas nos nós

Alocação Dinâmica: Pode ser mudada durante o processamento (migração de tarefas)

Alocação Estática: Eficiente para tarefas de comportamento previsível (e.g., periódicas), sobretudo tarefas críticas

Alocação ótima: Problema NP-Árduo

→ Para n tarefas e m processadores, existem m^n alocações possíveis

→ Uso de metaheurísticas

e.g., *Escalonamento global estático de tarefas periódicas*, usando

Annealing Simulado, *Annealing* de Tsallis, Busca Tabu e Otimização Microcanônica

- Modelo de Sistema Distribuído

Barramento com oito nós; capacidade de processamento e memória conhecidas

Protocolo de comunicação via passagem de ficha (*token-passing protocol*)

43 tarefas periódicas a serem alocadas; 5 tarefas críticas replicadas (tolerância a falhas)

Comunicação entre processadores: limite de tempo dado pelo tempo de circulação de ficha (*token rotation time, TRT*)

- Requisitos para as alocações

1. Restrições temporais de execução de cada tarefa devem ser satisfeitas (*escalonabilidade*)

2. Algumas tarefas requerem processadores específicos

3. Réplicas devem ser alocadas a processadores distintos

4. Tarefas devem respeitar a capacidade de processamento e de memória dos seus nós

5. O TRT deve ser o menor possível

- Função Custo

Cada requisito contribui um fator de penalidade, se não satisfeito

$$E_{total}(\Omega) = \sum_{i=1}^{n} K_i E_i$$

$\Omega \equiv$ alocação

$E_i \equiv$ penalidade associada ao requisito i

$K_i \equiv$ fator de ponderação associado ao requisito i

- Medidas de Desempenho

$T_H \equiv$ Tempo médio de execução ($segundos$)

$P_F \equiv$ Porcentagem das alocações que satisfazem todos os requisitos

$E_F \equiv$ Custo médio das alocações que satisfazem todos os requisitos

$T_{TRT} \equiv$ TRT médio das alocações que satisfazem todos os requisitos ($milissegundos$)

$P_{Min} \equiv$ Porcentagem das alocações que correspondem ao ótimo global