Alocarea Optimă a Resurselor în Sisteme Distribuite

O Abordare Microeconomică

Damian Alexandru Horneț Alex-Andrei Opran Andrei

Sisteme Distribuite
Conf. Dr. Andrei Pătrașcu

Cuprins

Problema Alocării Fișierelor (FAP)

Context

- Sisteme distribuite = agenți de calcul care partajează resurse
- Resurse divizibile: fișiere mari, baze de date
- Obiectiv: **maximizarea performanței**

Abordare Microeconomică

- Fiecare nod = agent economic
- Decizii bazate pe cost/beneficiu
- Negociere descentralizată pentru echilibru global

Parametrii Sistemului

Variabile:

- N = număr de noduri
- λ_i = rata de sosire cereri
- μ = rata de servire
- x_i = fracțiune din resursă
- K = cost comunicare

Constrângeri:

- $\sum_{i=1}^{N} x_i = 1$
- $x_i \geq 0, \forall i$
- $\mu > \lambda \cdot x_i$

Funcția de Cost

$$C = \sum_{i=1}^{N} (C_i + K \cdot T_i)\, \lambda_i, \quad T_i = \frac{1}{\mu - \lambda \cdot x_i}$$

Ciclul de Optimizare

1. **Calcul Local:** Fiecare nod calculează utilitatea marginală U_i'
2. **Comunicare:** Nodurile schimbă informații despre U_i'
3. **Actualizare:** Ajustarea alocării x_i pe baza diferențelor

3 Algoritmi Implementați

- Prima Derivată (Gradient Descent)
- A Doua Derivată (Newton)
- Pairwise Interaction

1. Algoritmul Prima Derivată

Caracteristici

- Gradient descent clasic
- Pas fix de învățare: $\alpha = 0.01$
- Actualizare: $\Delta x_i = -\alpha \cdot (d_{avg} - d_i)$

Avantaje:

- Simplu și stabil
- Convergență garantată

Dezavantaje:

- Convergență lentă
- Necesită multe iterații ($\sim$1500)

2. Algoritmul A Doua Derivată

Caracteristici

- Metoda Newton - folosește curbura
- $k_i = \frac{1}{\frac{\partial^2 U}{\partial x_i^2}}$ (factor de scalare)
- Actualizare: $\Delta x_i = -\alpha \cdot k_i \cdot (d_{avg} - d_i)$

Avantaje:

- Convergență mai rapidă
- Pași adaptivi

Performanță:

- $\sim$1000 iterații
- 3$\times$ mai rapid decât prima derivată

3. Algoritmul Pairwise Interaction

Caracteristici

- **Complet descentralizat**
- Comunicare doar între vecini
- Topologie definită: $\{(i, j)\}$ muchii
- Actualizare: $\Delta x_i = -\alpha \cdot \frac{k_i k_j}{k_i + k_j} \cdot (d_i - d_j)$

Avantaje:

- Cel mai rapid (<50 iterații)
- Scalabil
- Fără sincronizare globală

Observații:

- Depinde de topologie
- Ideal pentru sisteme mari

Comparație Algoritmi

Algoritm	**Iterații**	**Viteză**	**Comunicare**
Prima Derivată	$\sim$1500	$\times$	$O(N)$ global
A Doua Derivată	$\sim$1000	$\times\times\times$	$O(N)$ global
Pairwise	$<$50	$\times\times$	$O(grad)$ local

Tabela: Tabel pentru compararea performanțelor

Observații

- Toți converg către același cost optim
- Pairwise: cel mai rapid pentru sisteme mari
- A Doua Derivată: cel mai bun raport viteză/stabilitate

Grafic Convergență

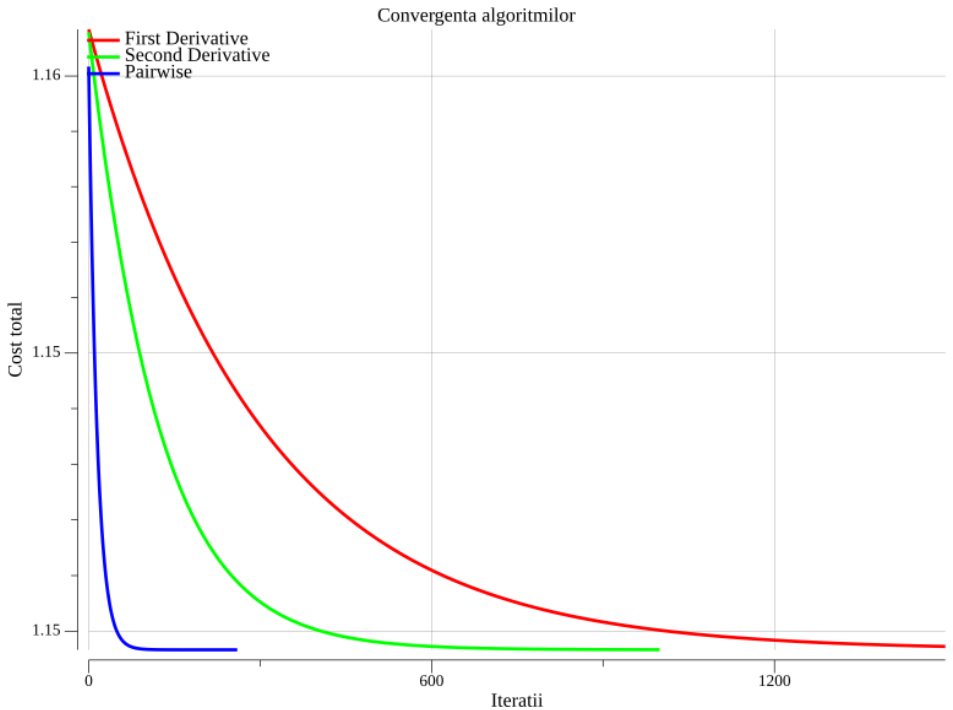

Figura: Evoluția costului pentru cei 3 algoritmi

- **Pairwise:** Convergență rapidă în < 50 iterații
- **A Doua Derivată:** Convergență lină în $\sim$1000 iterații
- **Prima Derivată:** Convergență treptată în $\sim$1500 iterații

Detalii Implementare

Tehnologii

- Limbaj: **Go** (concurrency nativ)
- Paralelizare: goroutines pentru calcule simultane
- Configurare: JSON (mu, lambdas, K)

Structura Nodului

- ID, Lambda, Mu, Allocation
- Metode: ComputeFirstDerivative, Compute1onSecondDerivative
- Normalizare și validare constrângeri

Concluzii

Realizări

- Implementare cu succes a 3 algoritmi descentralizați
- Validare experimentală a teoriei microeconomice
- Convergență către echilibrul Nash

Avantaje Abordare Descentralizată

- Scalabilitate superioară
- Toleranță la erori
- Minimizare comunicare globală (Pairwise)

Aplicații Practice

CDN-uri, cloud computing, baze de date distribuite, sisteme IoT

Mulțumim!

Repository: `alex6damian/File-Allocation-Problem`