Architetture e Programmazione dei Sistemi di Elaborazione
Progetto a.a. 2021/22

"Fish School Search"
in linguaggio assembly x86-32+SSE, x86-64+AVX e openMP

Fabrizio Angiulli Fabio Fassetti

1 Decrizione del problema.

La *Fish School Search (FSS)* è un algoritmo di ottimizzazione ispirato al comportamento dei banchi di pesci. L'idea di base dell'algoritmo nasce dai meccanismi di alimentazione e movimento coordinato tipico di questo contesto. In particolare, i pesci nuotano verso il "gradiente positivo" per poter mangiare e prendere così peso. Dato che i pesci più pesanti hanno maggiore influenza sul banco nel processo di ricerca collettiva di cibo, il baricentro del banco tende a spostarsi verso i posti migliori dello spazio di ricerca.

Preliminari.

Data una funzione $f : \mathbb{R}^d \to \mathbb{R}$ a d variabili, l'obiettivo dell'algoritmo di ottimizzazione di interesse per questo progetto è trovare il valore di $\widehat{\mathbf{x}} \in \mathbb{R}^d$ per cui il valore di f è minimo, ossia tale che $f(\widehat{\mathbf{x}}) \leq f(\mathbf{x}) \forall \mathbf{x} \neq \widehat{\mathbf{x}}$, assumendo di non conoscere il gradiente di f.

Fish School Search.

Dato un insieme di n pesci, le loro posizioni $\mathbf{x}_i, i \in [1, n]$ sono vettori di $\mathbb{R}^d$ rappresentanti n possibili soluzioni del problema. L'FSS comprende 3 passi principali:

1. *movimento individuale*: durante questa fase ogni pesce esegue un movimento casuale, se la posizione assunta è migliore la mantiene altrimenti torna alla precedente;

2. *movimento istintivo*: durante questa fase i pesce sono attratti verso le zone più promettenti;

3. *movimento volitivo*: questa fase è responsabile della regolamentazione tra esplorazione e sfruttamento della ricerca.

Movimento individuale.

Ogni pesce esegue una ricerca locale alla ricerca di regioni promettenti nello spazio di ricerca. In particolare, ogni pesce i valuta la nuova posizione

$$\mathbf{y}_i(j) = \mathbf{x}_i(j) + rand(-1, 1) \cdot step_{ind}, j \in [1, d], \tag{1}$$

dove $rand(-1, 1)$ indica un numero casuale compreso tra -1 e 1 e $step_{ind}$ controlla il passo di movimento individuale. Si noti che le coordinate vengono aggiornate separatamente, ossia ad ogni coordinata viene applicato un passo casuale indipendente.
Se la nuova posizione è migliore della precedente il pesce la mantiene altrimenti torna indietro. Viene quindi calcolata la variazione della funzione e della posizione come

$$\Delta f_i = f(\mathbf{y}_i) - f(\mathbf{x}_i)$$
$$\mathbf{\Delta x}_i = \mathbf{y}_i - \mathbf{x}_i$$

ovviamente qualora il pesce decidesse di mantenere la posizione precedente entrambe le variazioni sarebbero pari a 0. Si noti che essendo $\mathbf{\Delta x}_i$ pari alla differenza tra due vettori d dimensionali è anch'esso un vettore a d dimensioni.

Operatore di alimentazione.

Dopo il movimento individuale, viene applicato l'operatore di *alimentazione*, ossia ogni pesce i aggiorna il proprio peso, applicando l'equazione:

$$W_i = W_i + \frac{\Delta f_i}{\max_{j=1}^{n} \Delta f_j}. \tag{2}$$

Movimento istintivo.

In questa fase i pesci che hanno incontrato un maggiore miglioramento attirano i pesci nella propria posizione. In particolare, ogni pesce è incoraggiato a muoversi secondo l'equazione:

$$\mathbf{x}_i = \mathbf{x}_i + \boldsymbol{I} \tag{3}$$

dove

$$\boldsymbol{I} = \frac{\sum_{i=1}^{n} \Delta \mathbf{x}_i \cdot \Delta f_i}{\sum_{i=1}^{n} \Delta f_i},$$

è un vettore d dimensionale.

Movimento volitivo.

In questa fase viene regolata la capacità di esplorazione rispetto alla capacità di sfruttamento durante la ricerca. Per questo scopo, è necessario calcolare il peso totale del banco, il baricentro del banco e, inoltre, ogni pesce deve misurare la distanza tra la sua posizione e tale baricentro. Se il banco ha incrementato il proprio peso totale il banco si contrae verso il baricentro, ossia ogni pesce tende a muoversi verso tale punto per esplorare l'area. Viceversa, se il peso totale non incrementa, è necessario esplorare nuove aree. Quindi ogni pesce si allontana dal barientro e conseguentemente il banco si espande.
In dettaglio, viene calcolato il baricentro del banco

$$\mathbf{B} = \frac{\sum_{i=1}^{N} \mathbf{x}_i \cdot W_i}{\sum_{i=1}^{N} W_i} \tag{4}$$

in base alla posizione e al peso dei pesci.
Il peso totale del banco è $\sum_{i=1}^{N} W_i$ e ogni pesce si muove, ossia aggiorna la sua posizione, in accordo all'equazione:

$$x_i = x_i \pm step_{vol} \cdot rand(0,1) \cdot \frac{\mathbf{x}_i - \mathbf{B}}{dist(\mathbf{x}_i, \mathbf{B})}, \tag{5}$$

dove $dist(\cdot,\cdot)$ rappresenta la distanza euclidea, $rand(0,1)$ è un valore casuale compreso tra 0 e 1 e $step_{vol}$ è un valore che controlla lo spostamento.
In particolare, se il peso totale del banco è aumentato rispetto al valore precedentemente assunto, i pesci sono attratti verso il baricentro (segno − nell'equazione), altrimenti si allontanano (segno + nell'equazione).

Aggiornamento parametri.

I parametri $step_{ind}$ e $step_{vol}$ hanno un decadimento lineare nelle varie iterazioni. In formula:

$$step_{ind} = step_{ind} - \frac{step_{ind}(initial)}{It_{\max}}, \tag{6}$$

$$step_{vol} = step_{vol} - \frac{step_{vol}(initial)}{It_{\max}} \tag{7}$$

dove $step_{ind}(initial)$ e $step_{vol}(initial)$ sono i valori iniziali dei parametri e $It_{\max}$ è il numero di iterazioni del processo di ricerca.

Algoritmo.

L'algoritmo 1 riassume lo pseudocodice della metodologia. Per semplicità, si assuma che la funzione f abbia un solo minimo.

ALGORITMO 1: Fish School Search

Input: una funzione $f : \mathbb{R}^d \to \mathbb{R}$ e i parametri n_p, $step_{ind}$, $step_{vol}$, W_{scale} e $It_{\max}$
Output: il punto di minimo $\widehat{\mathbf{x}}$ di f

```
1  begin
2      foreach pesce do
3          Inizializza la posizione x_i con d valori casuali
4          Inizializza il peso W_i a W_scale / 2
5      while it < It do
6          foreach pesce do
7              Esegui il movimento individuale, Eq. (1)
8          Applica l'operatore di alimentazione, Eq. (2)
9          Esegui il movimento istintivo, Eq. (3)
10         Calcola il baricentro, Eq. (4)
11         Esegui il movimento volitivo, Eq. (5)
12         Aggiorna i parametri, Eq. (6)
       // Restituisci la posizione x̂ del pesce associata al minor valore di f
13     return x̂ = arg max f(x_i).
```

2 Descrizione dell'attività progettuale.

Obiettivo del progetto è mettere a punto un'implementazione dell'algoritmo Fish School Search in linguaggio C e di migliorarne le prestazioni utilizzando le tecniche di ottimizzazione basate sull'organizzazione dell'hardware.
L'ambiente sw/hw di riferimento è costituito dal linguaggio di programmazione C (`gcc`), dal linguaggio assembly x86-32+SSE e dalla sua estensione x86-32+AVX (`nasm`) e dal sistema operativo Linux (`ubuntu`).
In particolare il codice deve consentire di trovare il minimo di una funzione f con l'algoritmo 1, dati i valori dei parametri: `-np <`n_p`>`, `stepind <`$step_{ind}$`>`, `stepvol <`$step_{vol}$`>`, `Wsc <`W_{scale}`>` e `iters <`$It_{\max}$`>`.
Per semplicità, la funzione da minimizzare f sia della forma:

$$f(\mathbf{x}) = e^{\mathbf{x}^2} + \mathbf{x}^2 - \mathbf{c} \circ \mathbf{x},$$

dove $\mathbf{x}^2$ è il prodotto scalare di $\mathbf{x}$ con se stesso, $\mathbf{c} \circ \mathbf{x}$ indica il prodotto scalare di $\mathbf{c}$ e $\mathbf{x}$, e $\mathbf{c}$ è un vettore di d coefficienti passato in input. Quindi, ad esempio, se $d = 2$ e $c = [3, 5]$, la funzione f è

$$f\big(\mathbf{x} = [x(0), x(1)]\big) = e^{x(0)^2 + x(1)^2} + x(0)^2 + x(1)^2 - 3 \cdot x(0) - 5 \cdot x(1).$$

Per quanto riguarda la generazione di numeri random, essendo il numero di valori casuali necessari all'algoritmo pari a $l = It_{\max} \cdot (n_p \cdot (d + 1))$, viene fornito in input un vettore di l numeri random compresi tra 0 e 1 denominato

`rand_<d>_<np>_<iters>.ds2`,

dove $iters = It_{\max}$.
Quindi la chiamata avrà la seguente struttura:

```
./fss_<arch> -c <c> -r <r> -x <x> -np <np> -si <stepind> -sv <stepvol> -w <wscale> -it <itmax>
    <arch>: architettura associata all'eseguibile, {32c, 64c, 32ompc 64ompc}
    <c>: file ds2 contenente un vettore d dimensionale di coefficienti
    <r>: file ds2 contenente un vettore di numeri casuali
    <x>: file ds2 contenente le posizioni inziali dei pesci
    <np>: numero di pesci
    <stepind>: valore iniziale del parametro per il movimento individuale
    <stepvol>: valore iniziale del parametro per il movimento volitivo
    <wscale>: valore iniziale del peso
    <itmax>: numero di iterazioni
```

e sorgenti ed eseguibili devono essere contenuti in una cartella il cui nome è l'id del gruppo. Qualora un valore di un parametro (sia esso di default o specificato dall'utente) non sia applicabile, il codice deve segnalarlo con un messaggio e terminare.

Di seguito si riportano ulteriori linee guida per lo svolgimento del progetto:

- Si consiglia di affrontare il progetto nel seguente modo:
    1. Codificare l'algoritmo interamente in linguaggio C, possibilmente come sequenza di chiamate a funzioni;
    2. Sostituire le funzioni scritte in linguaggio ad alto livello che necessitano di essere ottimizzate con corrispondenti funzioni scritte in linguaggo assembly.

Ciò consentirà di verificare che l'algoritmo che si intende ottimizzare è corretto e di gestire più facilmente la complessità del progetto.

- Al fine di migliorare la valutazione dell'attività progettuale è preferibile presentare nella relazione un confronto tra le prestazioni delle versioni intermedie, ognuna delle quali introduce una particolare ottimizzazione, e finale del codice. Obiettivo del confronto è sostanziare la bontà delle ottimizzazioni effettuate.

- Occorre lavorare in autonomia e non collaborare con gli altri gruppi. Soluzioni troppo simili riceveranno una valutazione negativa. I progetti verranno messi in competizione.

- Sono richieste due soluzioni software, la prima per l'architettura x86-32+SSE e la seconda per l'architettura x86-64+AVX.

  Per i dettagli riguardanti la redazione del codice fare riferimento al file sorgente *c* `fss32c.c`, al file sorgente *nasm* `fss32.nasm`, allo script eseguibile `runfss32` (versione `x86-32+SSE`) e al file sorgente *c* `fss64c.c`, al sorgente *nasm* `fss64.nasm`, allo script eseguibile `runfss64` per la versione `x86-64+AVX` disponibili sulla piattaforma.

  Per l'interfacciamento tra programmi in linguaggio C e programmi in linguaggio assembly fare riferimento al documento allegato alla descrizione del progetto.

- È inoltre richiesta per ogni soluzione, una versione che faccia uso delle istruzioni `OpenMP`. I nomi dei relativi file dovranno contenere il suffisso "`_omp`" (es. `fss32_omp.c`).

- Il software dovrà essere corredato da una relazione. Per la presentazione del progetto è possibile avvalersi di slide.

- Prima della data di consegna del progetto verranno pubblicate le convenzioni da rispettare riguardanti i nomi e la collocazione dei file/directory al fine della compilazione e l'esecuzione dei codici di programma mediante script appositamente predisposti. **Dato l'elevato numero di progetti, sarà cura del candidato accertarsi di aver rispettato pienamente le convenzioni di consegna.**

Buon lavoro!