Parallel Processing in Financial Engineering

Sungjoo Ha
sungjooha@soar.snu.ac.kr
Seoul National University, Optimization and Financial Engineering Lab

Finding Attractive Technical Patterns

- Attractive technical patterns in stock market
  - Profitable
  - Human interpretable
  - Frequent
- Use genetic programming (GP) to evolve attractive technical patterns

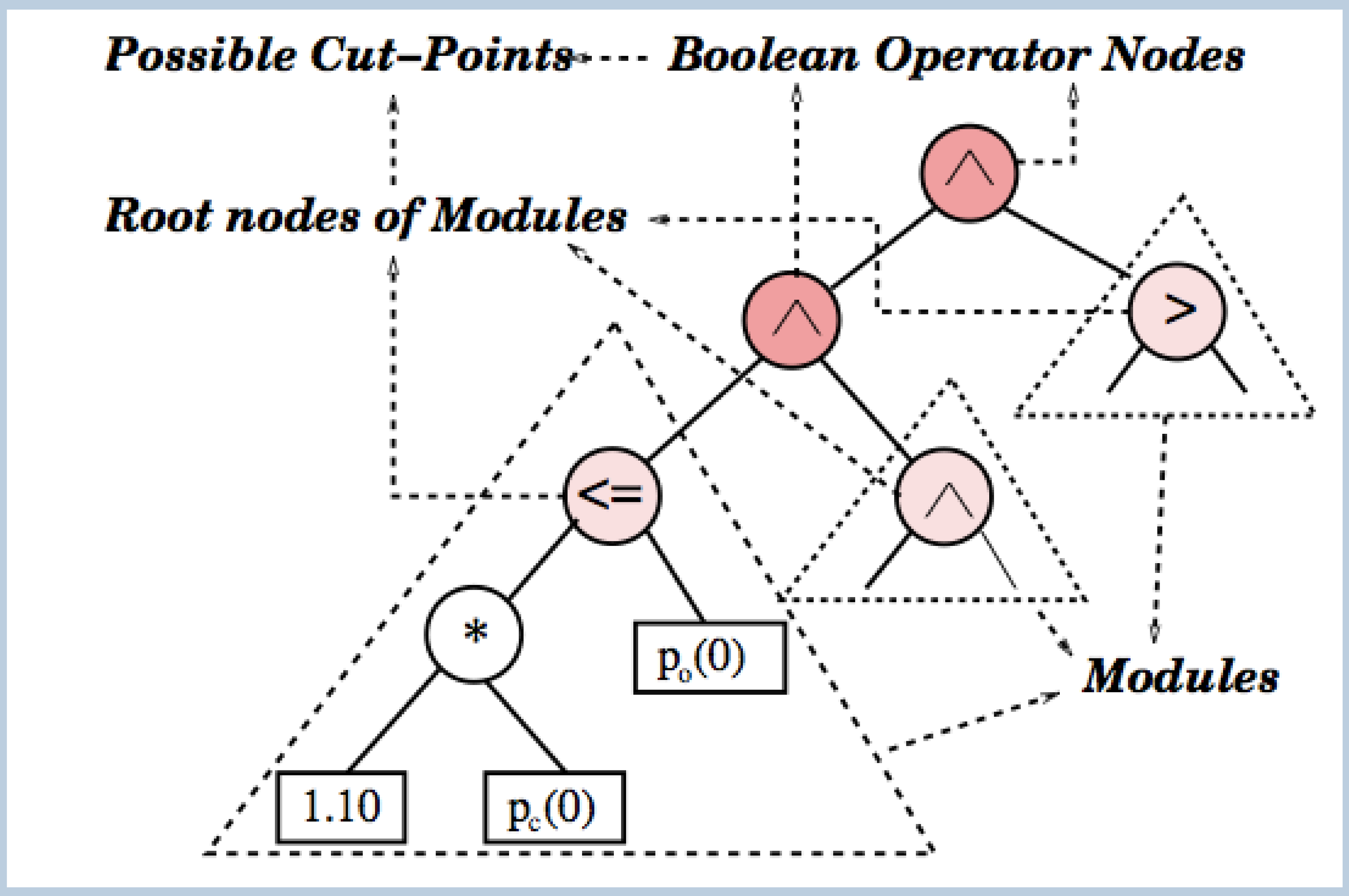

Figure 1: Genetic programming for finding attractive technical patterns

$$E_k(r) = \frac{1}{|R(r)|} \sum_{(i,j)\in R(r)} \frac{P_c(i, j+k)}{P_c(i,j)}$$

Figure 2: Expected earning rate of pattern r after k trading days

$$f(r) = \begin{cases} \frac{1}{n}\sum_{k=1}^{n} E_k(r) & \text{if } |r| < M, |R(r)| \geq m \\ 0 & \text{otherwise} \end{cases}$$

Figure 3: Fitness function for modular GP

Parallelization of Fitness Evaluation

- Fitness evaluation takes the majority of GP running time
  - But the evaluation of fitness function is embarrassingly parallel
  - Exploit this parallelism using GPGPUs
- Currently achieves more than 100 fold increase in processing power[a]
  - Multiple GPU devices[b]
  - Minimizing the data transfer between CPU and GPU
    * Load the data once
    * Only transfer required results
  - Using parallel algorithmic patterns
    * Prefix sum
    * Parallel reduction
- Future works
  - Explore sampling
  - Exploit more parallel algorithmic patterns

[a] Compared to single core version of the program
[b] NVIDIA GTX 690

Log-Optimal Portfolio Selection

- Maximizing the expected log investment return of a portfolio

$$\boldsymbol{b} = (b_1, b_2, \cdots, b_m)^t, \quad b_i \geq 0, \quad \sum b_i = 1$$
$$\boldsymbol{X} = (X_1, X_2, \cdots, X_m)^t \sim F(\boldsymbol{x}), \quad \boldsymbol{x} \in \boldsymbol{R}^m$$

Figure 4: A portfolio and stock vector

$$\begin{aligned} W(\boldsymbol{b}) &= E \ln \boldsymbol{b}^t \boldsymbol{X} = \int \ln \boldsymbol{b}^t \boldsymbol{x} \, \mathrm{d}F(\boldsymbol{x}) \\ W^* &= \max_b W(\boldsymbol{b}) \end{aligned}$$

Figure 5: Expected log investment return of a portfolio

- We wish to find an optimal portfolio $\boldsymbol{b}$
  - Various approaches are available
- Iterative algorithms approach
  - Need to calculate $W(\boldsymbol{b})$ but requires complicated integration
  - Can be overcome by the method of sampling

$$X_1, \ldots, X_n \sim p \quad \text{i.i.d.}$$
$$\hat{\mu}_n = \frac{1}{n}\sum_{i=1}^{n} f(X_i)$$

Figure 6: A basic Monte Carlo estimate of $Ef(X)$

Parallelization of Computing Expected Return

Do the following for each thread;
repeat
  Sample $\boldsymbol{X}$ from $F(\boldsymbol{x})$;
  Calculate $\ln \boldsymbol{b}^t \boldsymbol{X}$;
  Perform parallel reduction;
until *Some accuracy criteria*;

Algorithm 1: Parallel computation of $W(\boldsymbol{b})$

- Computing $W(\boldsymbol{b})$ fast enough allows us to use iterative methods for the optimization

Challenges

- Finding parallelizable component of a program
- Speed and accuracy trade off
  - Single precision vs. double precision
  - More complicated than single core environment
  - May even lead to better performance and better accuracy all at the same time
- Parallelization itself is still tedious and difficult
  - Carefully planning memory access pattern
  - Exploiting the parallel memory architecture
  - Concise representation of data
  - Host/device memory data transfer pattern
  - Exploiting parallel program patterns
  - Fine tuning using device level knowledge