Решение оптимизационных задач на распределенных вычислительных системах с помощью алгоритма Асинхронной Дифференциальной Эволюции

Е. Жабицкая, М. Жабицкий

Объединенный институт ядерных исследований, Дубна
Университет Дубна

MPAMCS, Дубна, 22-27 августа 2012

Содержание

Нахождение глобального минимума

Поиск вектора $\boldsymbol{x}^* = \{x_j\}|_{j=0,\dots D-1}$, минимизирующего целевую функцию $f(\boldsymbol{x})$:

$$f(\boldsymbol{x}^*) \leqslant f(\boldsymbol{x}), \quad \forall \boldsymbol{x} \in \Omega,$$

где Ω — область поиска.

«Отягчающие» особенности некоторых задач:

- Ограничения на параметры $\varphi(\boldsymbol{x}) < 0$
- Многопараметрические задачи $D = 10 \dots 100$
- Многомодальные целевые функции
- Недифференцируемые целевые функции
- Целевые функции, требующие значительных вычислений

Классическая Дифференциальная эволюция

- Дифференциальная эволюция (ДЭ) — эволюционный алгоритм со специфическим оператором *мутации*
- Разработан Р. Шторном и К.В. Прайсом в 1995
  [K. Price, R. Storn// J. Global of Optimization 11 (1997) 341]
- Оперирует *популяцией* векторов размером N_p
- Каждый член популяции — вектор в пространстве параметров $\Omega = R^D$

[K. Price, R. Storn, J.A. Lampinen "Differential evolution — A Practical Approach to Global Optimization", Springer, 2005]
[S. Das, P.N. Suganthan// IEEE Trans. Evol. Comp. 15 (2011) 4]

Классическая ДЭ: Алгоритм

// инициализация популяции $\{\boldsymbol{x}_{i,g=0}\}|_{i=0,\ldots N_{\boldsymbol{p}}-1}$, $\boldsymbol{x}_{i,g} = \{x_{i,j,g}\}|_{j=0,\ldots D-1}$

do {

for $(i = 0;\ i < N_p;\ i = i + 1)$ {

// Мутация:

$\boldsymbol{v}_{i,g} = \boldsymbol{x}_{r,g} + F(\boldsymbol{x}_{p,g} - \boldsymbol{x}_{q,g})$ // мут. вектор, $r \neq p \neq q$ — сл. индексы

// Кроссовер (рекомбинация):

for $(j = 0;\ j < D;\ j = j + 1)$

$$u_{i,j} = \begin{cases} v_{i,j} & \text{rand}(0,1) < C_r \text{ or } j = j_{\text{rand}} \\ x_{i,j,g} & \text{иначе} \end{cases}$$ // пробный вектор

// Отбор:

$$\boldsymbol{x}_{i,g+1} = \begin{cases} \boldsymbol{u}_i & \text{if } (f(\boldsymbol{u}_i) < f(\boldsymbol{x}_{i,g})) \\ \boldsymbol{x}_{i,g} & \text{иначе} \end{cases}$$

} $g = g + 1$; // переход к следующему поколению

} while (пока не выполнен критерий остановки);

Классическая ДЭ: синхронный алгоритм

// инициализация популяции $\{\boldsymbol{x}_{i,g=0}\}|_{i=0,\ldots N_p-1}$, $\boldsymbol{x}_{i,g} = \{x_{i,j,g}\}|_{j=0,\ldots D-1}$

do {

for $(i = 0;\ i < N_p;\ i = i + 1)$ {

// Мутация:

$\boldsymbol{v}_{i,g} = \boldsymbol{x}_{r,g} + F(\boldsymbol{x}_{p,g} - \boldsymbol{x}_{q,g})$ // мут. вектор, $r \neq p \neq q$ — сл. индексы

// Кроссовер (рекомбинация):

for $(j = 0;\ j < D;\ j = j + 1)$

$$u_{i,j} = \begin{cases} v_{i,j} & \text{rand}(0,1) < C_r \text{ or } j = j_{\text{rand}} \\ x_{i,j,g} & \text{иначе} \end{cases}$$ // пробный вектор

// Отбор:

$$\boldsymbol{x}_{i,g+1} = \begin{cases} \boldsymbol{u}_i & \text{if } (f(\boldsymbol{u}_i) < f(\boldsymbol{x}_{i,g})) \\ \boldsymbol{x}_{i,g} & \text{иначе} \end{cases}$$

} $g = g + 1$; // переход к новому поколению

} while (пока не выполнен критерий остановки);

Асинхронная Дифференциальная эволюция

// инициализация популяции $\{\boldsymbol{x}_i\}|_{i=0,\dots N_p-1}$, $\boldsymbol{x}_i = \{x_{i,j}\}|_{j=0,\dots D-1}$
do {

$i = choose\_target\_vector()$; // выбор целевого вектора i

// Мутация:

$\boldsymbol{v}_i = \boldsymbol{x}_r + F(\boldsymbol{x}_p - \boldsymbol{x}_q)$; // мут. вектор, $r \neq p \neq q$ — сл. индексы

// Кроссовер (рекомбинация):

for $(j = 0;\ j < D;\ j = j + 1)$

$$u_{i,j} = \begin{cases} v_{i,j} & \mathrm{rand}(0,1) < C_r \text{ or } j = j_{\mathrm{rand}} \\ x_{i,j} & \text{иначе} \end{cases} \quad \text{// пробный вектор}$$

// Отбор:

if $(f(\boldsymbol{u}_i) < f(\boldsymbol{x}_i))$

$\boldsymbol{x}_i = \boldsymbol{u}_i$;

} while (пока не выполнен критерий остановки);

[E. Zhabitskaya, M. Zhabitsky// LNCS 7125, 328, 2012]

Адаптация размера популяции N_p

Параметры алгоритма: F, C_r; N_p

	малый N_p	большой N_p
Вероятность сходимости	−	+
Скорость сходимости	+	−

Алгоритм АДЭ с рестартом:

- Малый размер начальной популяции N_p
- Рестарт с увеличенным N_p, если диагностирована стагнация сходимости (длительное отсутствие прогресса или вырождение популяции)

Адаптация размера популяции в соответствии со сложностью решаемой проблемы!

Асинхронная ДЭ: классификация алгоритмов

мутация: $\mathbf{v}_i = \mathbf{x}_r + F(\mathbf{x}_p - \mathbf{x}_q)$

$\mathbf{x}_i$ — целевой вектор;
$\mathbf{x}_r$ — базовый вектор
$(\mathbf{x}_p - \mathbf{x}_q)$ — дифференциальный вектор

DE/w/x/y/z в соответствии с операторами Мутации и Кроссовера:
w отвечает заменяемому целевому вектору;
x отвечает базовому вектору;
y — число дифференциальных векторов;
z — тип кроссовера (биномиальный или экспоненциальный).

rand/rand/1/bin	$\mathbf{v}_{\text{rand}} = \mathbf{x}_{\text{rand}} + F(\mathbf{x}_p - \mathbf{x}_q)$
rand/best/1/bin	$\mathbf{v}_{\text{rand}} = \mathbf{x}_{\text{best}} + F(\mathbf{x}_p - \mathbf{x}_q)$
worst/best/bin	$\mathbf{v}_{\text{worst}} = \mathbf{x}_{\text{best}} + F(\mathbf{x}_p - \mathbf{x}_q)$

Асинхронная Дифф. эволюция: pros and cons

Преимущества:

- Найденный "улучшенный" вектор сразу же принимает участие в эволюции
- Новые типы стратегий
- Отсутствует барьерная синхронизация, типичная для КДЭ при смене поколений

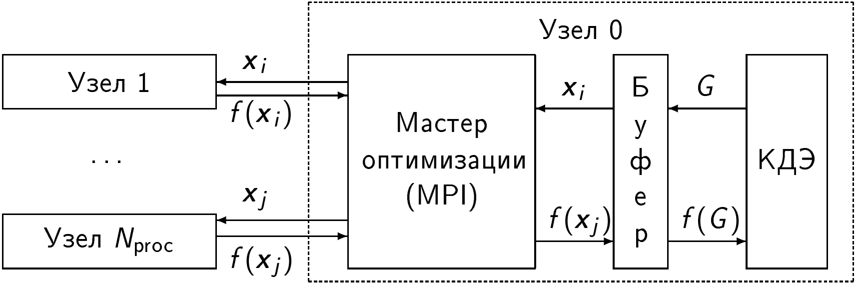
Недостатки:

- Бо́льшая вероятность вырождения (потеря *популяционного разнообразия*) [E. Zhabitskaya// LNCS 7125, 322, 2012]

Необходимо сравнить:

- Скорость и вероятность сходимости
- Ускорение при параллельных вычислениях

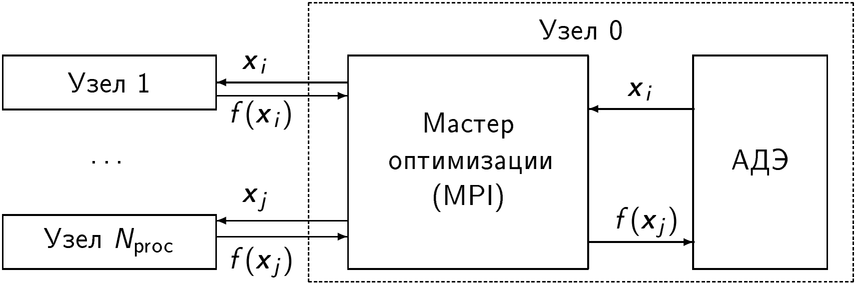
Параллельные расчеты на распределенных вычислительных системах (КДЭ)



[J. Lampinen, 1999]

- Master/Slave модель
- Потери выч. времени из-за барьерной синхронизации

Параллельные расчеты на распределенных вычислительных системах (АДЭ)



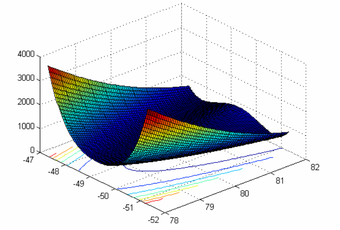
- Master/Slave модель
- Полная и эффективная! загрузка вычислительных узлов
- Мастер оптимизации в стандарте MPI (MPICH2)

Ускорение при параллельных вычислениях I

Смещенная функция Розенброка

$$f_6(\boldsymbol{x}) = \sum_{j=1}^{D-1} \left(100\left(z_j^2 - z_{j+1}^2\right) + (z_j - 1)^2\right) + f_{\text{bias}},$$

$$\boldsymbol{z} = \boldsymbol{x} - \boldsymbol{o} + 1, \quad \boldsymbol{x} \in [-100, 100]^D$$



Критерии оценки

- Вероятность сходимости

$$P_{\text{succ}} = \frac{N_{\text{succ}}}{N_{\text{trials}} = 25}$$

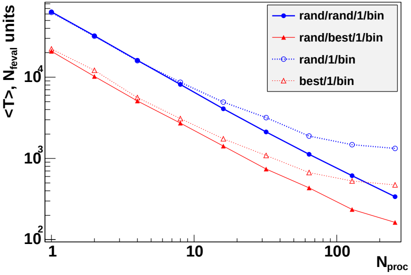
- Кол-во вызовов f (усредненное по сошедшимся попыткам)

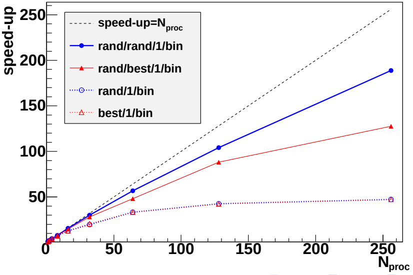
$$\langle N_{\text{feval}} \rangle = \frac{1}{N_{\text{succ}}} \sum_i (N_{\text{feval}})_i$$

Ускорение при параллельных вычислениях II

- Целевые функции, требующие значительных вычислений
- Каждому вычислению ф. Розенброка ставим в соответствие случайный временной интервал $t_k = N(1, 0.2)$
- $D = 10, \quad N_p = 40$

$$\text{speed-up}(N_{proc}) = \frac{\langle T(N_{proc} = 1)\rangle}{\langle T(N_{proc})\rangle}$$





Результаты

- Сформулирован алгоритм Асинхронной Дифференциальной эволюции
- Автоматический подбор размера популяции в соответствии со сложностью решаемой проблемы
- Лучшая параллельность по сравнению с классической Дифференциальной эволюцией
- Решение реальных задач