一个随机并行算法的收敛性分析*

乔长阁

（清华大学计算机科学与技术系）

CONVERGENCE ANALYSIS OF A STOCHASTIC PARALLEL ALGORITHM

Qiao Changge

(Department of Computer Science and Technology, Tsinghua University)

Abstract

As a stochastic parallel algorithm, Alopex has got good results in solving combinatorial optimization problems. In this paper, we use Bernoulli trials probability model to show convergence of the algorithm, and we have got true convergence result. The proof process can be used to guide the users to use the algorithm appropriately.

1. 引　　言

在科学技术、工程和商业领域中有许多组合优化问题，即对多自变量非凸函数求全局极值问题，绝大多数算法易陷入局部极值．1988年，Pandya[1]将一个随机并行算法Alopex用于解决一些组合优化问题和模式匹配问题，取得了很好的结果[2]．试验证明，它可广泛用于各种组合优化问题的求解上．

作为一种启发式与随机优化相结合的算法，Alopex既克服了传统启发式算法中易陷入局部极值的缺陷，也克服了模拟退火算法中从随机(完全)搜索到梯度搜索收敛极为缓慢的不足．它通过从前次自变量变化对目标函数产生的影响中得到启发，用过程控制参数“温度”来控制概率大小，从而完成整个过程的优化．它利用“噪声”来摆脱局部极值．

Pandya在将Alopex算法用于组合优化问题时，对算法的收敛性进行了论证，但由于他采用了错误的古典概型，所以收敛性证明不正确．作者在利用Alopex算法的基础上[3]．对其收敛性进行了研究，并利用贝努利试验概型[4]对算法的收敛性进行了有效的证明，对今后算法的应用将有直接的指导意义．

* 1995年6月15日收到.

2. Alopex 优化算法及 Pandya 的收敛性论证

对实际优化问题，常可归结为一个求目标函数 $F(x_1, x_2, \cdots, x_N)$ 全局极值的问题，其中 $x_1, \cdots, x_N$ 为待确定的变量. Alopex 算法为

$$x_i(t)=x_i(t-1)+\delta_i(t), \tag{1}$$

$$\delta_i(t)=\begin{cases} +\delta, & 概率\ p_i(t), \\ -\delta, & 概率\ 1-p_i(t), \end{cases} \tag{2}$$

$$p_i(t)=\frac{1}{1+e^{(\pm\Delta_i(t)/T)}}, \tag{3}$$

$$\Delta_i(t)=[x_i(t-1)-x_i(t-2)]*[F(t-1)-F(t-2)], \tag{4}$$

式中 $F(x_1, x_2, \cdots, x_N)$ 为目标函数，$x_i(t)$ 为 F 中第 i 个自变量在 t 时刻的取值，$\delta_i(t)$ 为 t 时刻自变量 x_i 随机行走的步长，$p_i(t)$ 为 x_i 在 t 时刻向增加 δ 方向行走的概率. (3) 中正、负号选取与实际问题有关：正号使 F 极小化，负号使 F 极大化. $F(t-1)$、$F(t-2)$ 为 F 在 $(t-1)$、$(t-2)$ 时刻的取值.

在将 Alopex 算法应用于某一个具体问题时加以确定式(1)—(4)中的参数 δ(变量变化步长)和温度 T. 一般说来，我们取 $T=|\Delta_i|$，而 δ 的大小根据变量变化的范围来确定，通常取为变量变化动态空间的几百分之一或更小. 变量的初始值可以任意选取，它对算法的收敛没有影响，只是影响其收敛速度.

这个算法的特点是所有自变量在每一时刻同时发生变化，有利于实现并行操作，从而提高算法的计算效率. 每一自变量“行走方向”由概率 $p_i(t)$ 决定，此 $p_i(t)$ 是受一个称为“温度”的参量 T 的控制，从而使算法具有一定爬坡能力. 同时，自变量变化受该自变量在前次变化后对 F 作用之牵制，因而使进程从前次变化中得到启发.

为便于描述，我们以极大化目标函数为例，此时(3)式变为
$p_i(t)=1/(1+\exp(-\Delta_1(t)/T))$. 下面先给出几个定义：

定义 1. 目标函数的变化 ΔF: $\Delta F(t)=F(t)-F(t-1)$，即 ΔF 为从 $t-1$ 时刻变化到 t 时刻目标函数 F 的变化量.

定义 2. 自变量的变化 Δx_i：$\Delta x_i(t)=x_i(t)-x_i(t-1)$，即 Δx_i 为从 $t-1$ 时刻变化到 t 时刻第 i 个自变量的变化量.

定义 3. 对于含有 N 个变量的目标函数 $F(x_1, x_2, \cdots, x_N)$，若 $\Delta F\geqslant 0$，则目标函数向其优化值趋近，称 F 正向变化，ΔF 是正确的；反之，$\Delta F<0$，称 F 反向变化，ΔF 是错误的.

定义 4. 对于目标函数 F，若自变量 x_i 的变化 Δx_i 使 F 正向变化，则称 Δx_i 是正确步长；反之称 Δx_i 是错误步长.

定义 5. 若变量 x_i 在前一时刻变化中，$\Delta x_i(t-1)$ 是正确步长，在此刻自变量 x_i 仍按原方向变化，则称 x_i 正向变化；反之称 x_i 反向变化.

不失一般性，假设 F 同等地依赖 N 个变量，即

$$\partial F / \partial x_i = \partial F / \partial x_j \ , \ i, \ j=1, \ 2, \ \cdots, \ N. \tag{5}$$

但必须强调，这并非算法成功所必需的条件.

Pandya 的证明是这样的[2]：对由(1)—(4)式给出的 Alopex 算法，考虑一个变量 x_i 的变化. 根据(2)和(3)两式，x_i 由趋向或背向 A_i 大小不变的步长 δ 组成，A_i 对应于 F 最佳值时 x_i 的取值. 在 N 个变量中，假设有一个变量 x_i 上一次变化步长 Δx_i 为正确步长，则余下的 $N-1$ 个变量的变化 $\Delta x_{j\neq i}$ 有 2^{N-1} 个不同的组合. 进一步假设 $N-1$ 为一个偶数，这 2^{N-1} 个状态落入三种情形：

情形 1. 具有大多数正确步长，在所有这些情况(n_1)下 ΔF 将为正确的.

情形 2. 具有相同数目的正确步长和错误步长，在所有这些情况(n_2)下 ΔF 将为正确的(因为已假定 Δx_i 为正确步长).

情形 3. 具有大多数错误步长，在所有这些情况(n_3)下 ΔF 将为错误的.

现在计算属于这三种情形的组态，可以通过估算第二种情形的数目来进行，余下的情形在第一种和第三种情形中均匀分配. 在 $N-1$ 个变量中，有$(N-1)/2$ 个正确步长和$(N-1)/2$ 个错误步长的情形数为

$$n_2 = C_{N-1}^{(N-1)/2} = (N-1)! \ / [(N-1)/2]!^2 . \tag{6}$$

总的情形数目为 $n_{all} = 2^{N-1}$，由于 $n_1 = n_3$，所以

$$n_1 = n_3 = (n_{all} - n_2)/2 = \{2^{N-1} - (N-1)! \ /[(N-1)/2]!^2\}/2 . \tag{7}$$

由此 Pandya 得出，在已知一个变量 Δx_i 为正确步长的条件下，ΔF 正确的概率为

$$\begin{aligned} P_N &= \Delta F \text{正确的情形数} / n_{all} = (n_1+n_2)/n_{all} \\ &= \{(N-1)! \ /[(N-1)/2]!^2 + [2^{N-1} - (N-1)! \ /[(N-1)/2]!^2]/2\}/2^{N-1} . \end{aligned} \tag{8}$$

$N=1$ 时 $P_N=1$，$N\_>\infty$ 时 $P_N=0.5$，当 N 很大时，(8)式可以用 Stirling 逼近，即

$$n! = e^{-n} n^n \sqrt{2\pi n} . \tag{9}$$

(8)式可化简为

$$P_N = 0.5 + 1/\sqrt{2\pi(N-1)} . \tag{10}$$

若假设 $T=|\Delta_i|$，则 Δx_i 为正确步长的概率为

$$\rho_c = 1/[1+\exp(-1)] = 0.73. \tag{11}$$

同样，Δx_i 为错误步长的概率为 $1-\rho_c$. x_i 正向变化的概率 ρ_N 可以用 P_N 与 ρ_c 来书写，即 x_i 取一个趋向 A_i 的阶跃的概率为两项之和：

$$\begin{aligned} \rho_N &= P(\Delta F \text{正确} |\Delta x_i \text{正确})^* P(\Delta x_i \text{正确}) + P(\Delta F \text{正确} |\Delta x_i \text{错误})^* P(\Delta x_i \text{错误}) \\ &= P_N {}^* \rho_c + (1-P_N)^* (1-\rho_c). \end{aligned} \tag{12}$$

因此，过程可描述为一个不受限制的随机游动，x_i 趋向或离开目标 A_i 的阶跃有不等概率 ρ_N 和 $1-\rho_N$. Pandya 因此认为，达到目标的概率 ρ_N(>0.5) 比错过目标的概率 $1-\rho_N$(<0.5) 大，经过多次迭代后 x_i 收敛至最佳值.

3. 利用贝努利试验论证 Alopex 算法的收敛性

可以很容易看出，上述收敛性证明不正确. 对于(12)式，

$$\partial\rho_N / \partial\rho_c = 2P_N - 1, \tag{13}$$

而由式(8)，$P_N = 0.5 + (N-1)! / \{2^N * [(N-1)/2]!^2\} > 0.5$，所以 $\partial\rho_N / \partial\rho_c > 0$，$\rho_N$ 为 ρ_c 的单调递增函数. 故此有

$$\rho_{\max} = \rho_N |_{\rho_c=1} = P_N \ . \tag{14}$$

当 N 很大时，用 Stiring 逼近有

$$\rho_{\max} = 0.5 + 1 / \sqrt{2\pi(N-1)}\ , \tag{15}$$

故 $0.5 < \rho_{\max} < (0.5 + 1/\sqrt{2\pi})$，$\rho_{\max}$ 无法趋近于 1. 因而无法保证当 $\rho_c = 1$(即自变量正向变化的概率为 1)时，算法由随机搜索转变为梯度搜索以使整个过程向目标函数优化值逼近. 可见，这样推导的结论不能说明 Alopex 算法是向目标函数的优化值收敛的.

实际上，上述证明忽视了概率统计中等可能概型(即古典概型)与 Bernoulli 概型的区别. 只有当样本空间所含的有限个基本事件发生的可能性相等时，才能有某一事件出现的概率等于该事件所含的有限个基本事件数目除以基本事件总数，即式(8)成立. 在 Alopex 算法中，通常 x_i 正向变化的概率不等于 x_i 反向变化的概率，因此(8)式采用古典概型推导出的 ΔF 正确的条件概率不成立.

事实上，对于 F 中的 N 个变量，当求 F 的全局极大值时，自变量 x_i 向增加 δ 方向变化的概率为 $\rho_c = 1/[1+\exp(-\Delta_i/T)]$，若它使得 ΔF 正向变化，则 x_i 正向变化的概率为 $\rho_c = 1/[1+\exp(-\Delta_i/T)]$，而反向变化的概率为 $1-\rho_c = 1/[1+\exp(\Delta_i/T)]$；反之，若自变量 x_i 向增加 δ 方向变化的概率为 $\rho_c = 1/[1+\exp(-\Delta_i/T)]$. 若它使得 ΔF 反向变化，则 x_i 反向变化的概率为 $\rho_c = 1/[1+\exp(-\Delta_i/T)]$，而正向变化的概率为 $1-\rho_c = 1/[1+\exp(\Delta_i/T)]$. 由(4)式的计算我们可以看到，无论 x_i 的变化方向如何，其正向变化的概率均为 $1/[1+\exp(-|\Delta_i|/T)]$，而反向变化的概率为 $1/[1+\exp(|\Delta_i|/T)]$，显然正向和反向变化的概率是不相等的.

在同一时刻，N 个变量的变化是相互独立的，故可视为 N 重 Bernoulli 试验. 因而，当有一半以上变量正向变化时由(5)式可知 ΔF 将为正确的. 如上所述，同样假设 $N-1$ 为偶数，当有 i 个变量正向变化时 F 正向变化的概率为 $C_N^i \rho_c^i (1-\rho_c)^{N-i}$，则 ΔF 正确的概率为

$$P(\Delta F) = C_N^N \rho_c^N (1-\rho_c)^0 + C_N^{N-1} \rho_c^{N-1} (1-\rho_c)^1 + \cdots$$
$$+ C_N^i \rho_c^i (1-\rho_c)^{N-i} + \cdots + C_N^{(N+1)/2} \rho_c^{(N+1)/2} (1-\rho_c)^{(N-1)/2} \ . \tag{16}$$

上式为当有 N 个，$N-1$ 个，…，$(N+1)/2$ 个变量正向变化时 F 正向变化的概率. 由于 $C_N^{N-i} = C_N^i$，上式对 ρ_c 求导有

$$dP(\Delta F) / d\rho_c = N\rho_c^{N-1} + (N-1) C_N^1 \rho_c^{N-2} (1-\rho_c)^1 - C_N^1 \rho_c^{N-1} + \cdots \ . \tag{17}$$

考虑上式第 i 项与第 $i+1$ 项的导数

$$(N-i)C_N^i\rho_c^{N-i-1}(1-\rho_c)^i-iC_N^i\rho_c^{N-i}(1-\rho_c)^{i-1}+(N-i-1)C_N^{i+1}\rho_c^{N-i-2}(1-\rho_c)^{i+1}$$
$$-(i+1)C_N^{i+1}\rho_c^{N-i-1}(1-\rho_c)^i,$$

由于$(N-i)C_N^i=(i+1)C_N^{i+1}$，可见第$i$项求导得到的正号项与第$i+1$项求导得到的负号项刚好抵消．故(17)式可简化为

$$dP(\Delta F)/d\rho_c=[(N+1)/2]^*C_N^{(N-1)/2}\rho_c^{(N+1)/2}(1-\rho_c)^{(N-1)/2}. \tag{18}$$

由于$0\leqslant\rho_c\leqslant1$，所以$dP(\Delta F)/d\rho_c\geqslant0$，故$P(\Delta F)$单调变化．

结论1．当$\rho_c=0.5$时，$P(\Delta F)=(1+C_N^1+\cdots+C_N^{(N-1)/2})/2^N=0.5$，即各自变量正向变化和反向变化的概率相等时，$F$正向变化和反向变化的概率也相等，此时$F$具有最大的随机性，"爬坡能力"最强．

结论2．当$\rho_c>0.5$时，$P(\Delta F)>0.5$，即目标函数向其优化值趋近的概率随自变量正向变化的概率的增加而单调增加，目标函数正向变化的概率大于反向变化的概率．

结论3．当$\rho_c=1$时，$P(\Delta F)=1$，目标函数F只向其优化值趋近，此时，整个过程已由随机变化转变成了以确定方向变化，即整个进程变成了梯度下降搜索．应当指出，当N为偶数时，同样可以得出上述结论．

由此可见，Alopex算法在对目标函数进行优化的过程中，可以通过控制自变量正向变化的概率ρ_c来控制整个搜索过程由随机搜索向梯度搜索的转化，并最终收敛至F的最优值上．另外我们还可看到，(5)式假设对算法的收敛性并无影响．从算法中我们看到，N个自变量的变化是相对独立的，且它们对目标函数的作用是相同的．如果$\partial F/\partial x_i\neq\partial F/\partial x_j$，则说明变量对目标函数的作用大小不同，因而只是在(16)式中项数上有些差异而已，但$P(\Delta F)$仍随ρ_c单调增加．

从上面的分析中不难看出，算法的整个进程为由随机搜索向梯度下降搜索的优化，它是通过自变量朝正向变化的概率ρ_c的大小来控制的．而ρ_c的大小却由被称为温度的参量T来控制，因此温度T的变化对进程影响深刻．初始时，我们可以选择一个较大的T，以使得$\rho_c\approx0.5$，这时系统处于一种随机状态，然后逐渐将T值减小直至最后使$T>0$，这时系统进入梯度搜索直至收敛至最佳值．

4. 结　论

本文采用BERNOULLI概型对一个随机并行算法的收敛性进行了证明，得到了正确的收敛结果．算法的收敛证明对通过选择合适的参数指导算法的收敛进程有积极的指导意义．

参　考　文　献

[1] Harth E. Pandya A. S., Dynamics of Alopex Process: Applications to Optimization Problems. in Biomathematics and Related Computational Problems. Kluwer Academic Publishers. 1988. 459—471.

[2] Pandya A. S., A Stochastic Parallel Optimization Algorithm. Ph. D. Dissertation. Syracuse University (U.S.A.). 1988.

[3] 乔长阁，高德远，一个随机并行算法及其在VLSI布图中的应用，西北工业大学学报，12(1) (1994) 74—78.

[4] 浙江大学数学系，概率论与数理统计，北京：高等教育出版社，1985年.