中国科学院软件研究所学术年会’2017
暨计算机科学国家重点实验室开放周

学术论文

基于分枝定界搜索的广义线性约束优化算法

陈力　吕荫润　吴敬征　王翀　张常有　Nasro Min-Allah　Jamal Alhiyafi　王永吉

ywang@itechs.iscas.ac.cn

Li Chen, Yinrun Lyu, Jingzheng Wu, Chong Wang, Changyou Zhang, Nasro Min-Allah, Jamal Alhiyafi, and Yongji Wang. Solving Linear Optimization over Arithmetic Constraint Formula. *Journal of Global Optimization*, published online, 2017.

受邀于2017年10月在运筹学和管理科学领域的顶级会议INFORMS (Institute for Operations Research and the Management Sciences) 年会上作报告

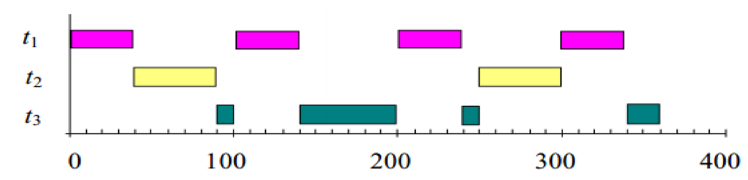
广义线性约束优化问题描述

$$\max \quad f(x) = a_1x_1 + a_2x_2 + \cdots + a_nx_n$$

$$\begin{aligned}
s.t. \quad & (g_{11}(\boldsymbol{x}) \le 0 \vee g_{12}(\boldsymbol{x}) \le 0 \cdots g_{1q_1}(\boldsymbol{x}) \le 0) \\
& \wedge (g_{21}(\boldsymbol{x}) \le 0 \vee g_{22}(\boldsymbol{x}) \le 0 \cdots g_{2q_2}(\boldsymbol{x}) \le 0) \\
& \cdots \\
& \wedge (g_{m1}(\boldsymbol{x}) \le 0 \vee g_{m2}(\boldsymbol{x}) \le 0 \cdots g_{mq_m}(\boldsymbol{x}) \le 0) \\
& (lb_1 - x_1 \le 0) \wedge (x_1 - ub_1 \le 0) \\
& \wedge (lb_2 - x_2 \le 0) \wedge (x_2 - ub_2 \le 0) \\
& \cdots \\
& \wedge (lb_n - x_n \le 0) \wedge (x_n - ub_n \le 0)
\end{aligned}$$

- 将传统运筹学优化问题的约束条件拓展为同时含有逻辑“与” “或”关系的等式/不等式组
- 与线性算数优化模理论问题具有等价的问题描述

应用场景一：实时系统单调速率优化设计



$$\max \quad f_{l_1l_2\cdots l_n} = \sum_{i=1}^{n} \frac{C_i}{T_i}$$

$$s.t. \quad \begin{cases} \sum_{j=1}^{i} \lceil p_{il_i}/T_j \rceil C_j - p_{il_i} \le 0 \\ C_i^{\min} \le C_i \le C_i^{\max} \end{cases}$$

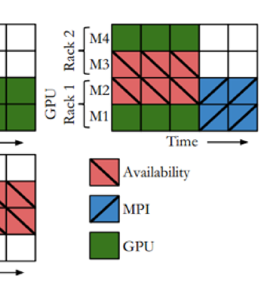
$$i = 1, 2, \cdots, n$$

应用场景二：异构集群资源调度

$$F(t) \triangleq (f = Budget \wedge t \le Desired) \vee (f = Penalty \wedge t > Deadline) \vee$$
$$(f = \frac{Budget - Penalty}{Desired - Deadline} t + \frac{Penalty * Desired - Budget * Deadline}{Desire - Deadline} \wedge t > Desired \wedge t \le Deadline)$$

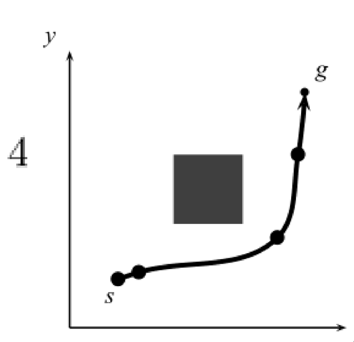
max(nCk({M1, M2, M3}, k=1, start=0, dur=20, v=1),
nCk({M1, M2, M3}, k=1, start=10, dur=20, v=1),
nCk({M1, M2, M3}, k=1, start=20, dur=20, v=1))

max(nCk({M1, M2, M3}, k=3, start=0, dur=10, v=1),
nCk({M1, M2, M3}, k=3, start=10, dur=10, v=1))



应用场景三：机器人路径规划

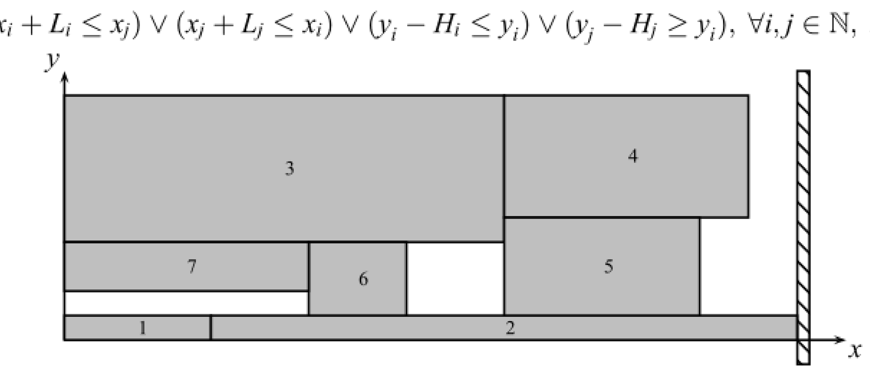
- 起点 $s = (0.7, 0.7)$，终点 $g = (8, 8)$
- 障碍物区域 $P = \{(x, y) | x \le 4 \wedge x \ge 2 \wedge y \ge 2 \wedge y \le 4\}$
- 目标函数 $f = (x-8)^2 + (y-8)^2$
- 迭代初始点 $p_0 = s$
- 约束条件：$x \ge 4 \vee x \le 2 \vee y \le 2 \vee y \ge 4$



应用场景四：切割剪裁/装箱问题

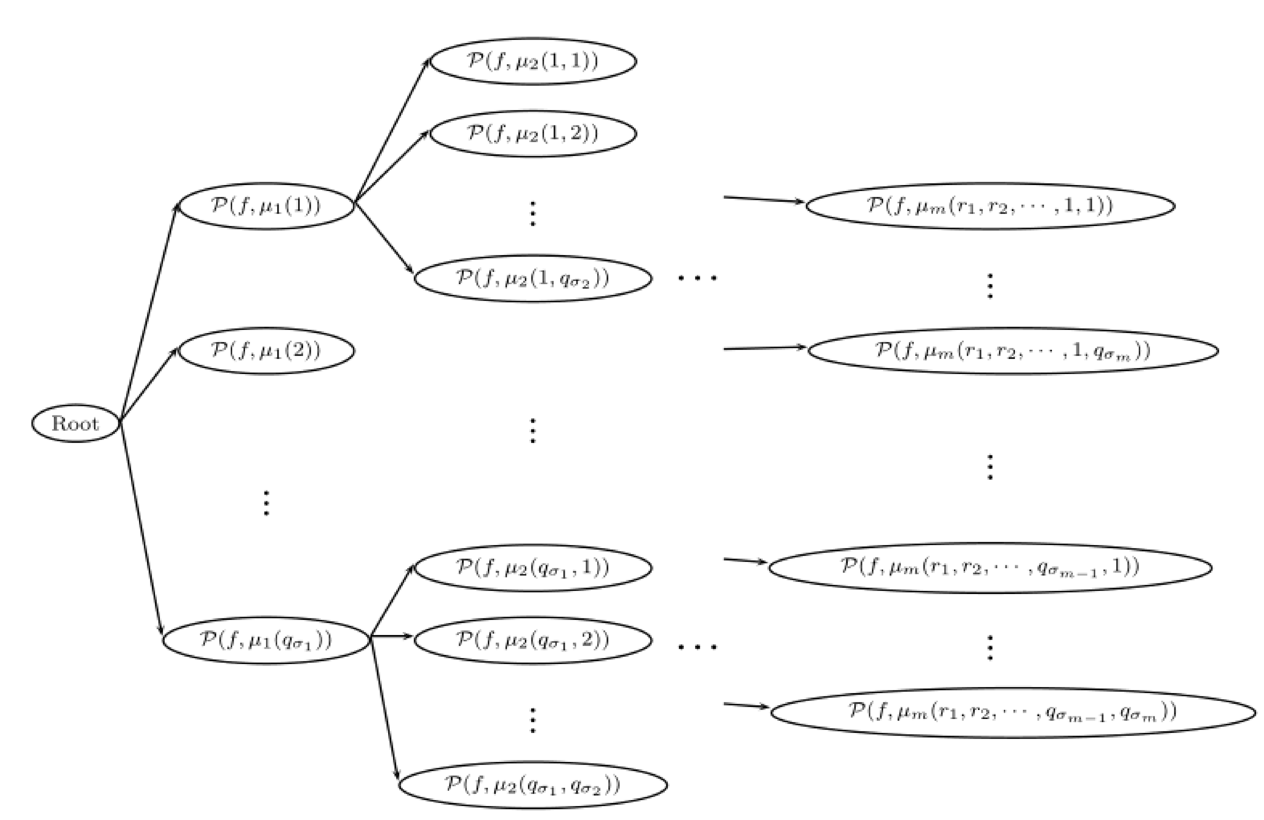
- 矩形 i：长 L_i，宽 H_i，左上角坐标 (x_i, y_i)
- 目标函数：板条长度 l
- 约束条件 (矩形的相互位置关系)：

$$(x_i + L_i \le x_j) \vee (x_j + L_j \le x_i) \vee (y_i - H_i \le y_j) \vee (y_j - H_j \ge y_i), \forall i, j \in \mathbb{N}, i < j$$



基本算法框架：

1. 将约束条件等价为若干个合取范式，将原问题分解为若干个线性规划子问题。每个子问题最优解的最大值为原问题的最优解。
2. 构造搜索树，每个叶节点表示一个分解后的线性规划子问题。每个节点相等当且仅当线性规划问题的约束条件相同；父节点的最优值不小于子节点的最优值。
3. 利用分枝定界思想，按深度优先搜索寻找最优解。若当前节点的线性规划子问题最优解不大于当前可行解，则将该节点剪枝。



Algorithm Improved algorithm of RS- LPT

Input:
$\widetilde{\mathcal{P}}$: the linear-arithmetic optimization problem to be solved
ε: a small positive number for measuring sampling
N_p: step size of sampling
N: sample size
TB: bound for counting visited nodes
Output:
$f^*_{\widetilde{\mathcal{P}}}$: the optimal solution of $\widetilde{\mathcal{P}}$
$x^*_{\widetilde{\mathcal{P}}}$: the maximizer of $f^*_{\widetilde{\mathcal{P}}}$

```
function FindOptimal
    for i ∈ I do
        Reduce(T_i)
    end for
    Decompose P̃ into 𝒫 = {P'|P' = P(f, λ(t)), ∀t ∈ 𝔗}.
    Construct a search tree where 𝒫_T is the set of linear programs in all the nodes, and 𝒫_L
    is the set of linear programs in the leaf nodes.
    (f_0, x_0) ← Sample(null, 𝒫_L, ε, N_p, N)
    count ← 0
    P_current ← P(f, μ_1(1))
    while Depth(P_current) ≠ 0 do
        (f_c, x_c) ←LPSolve(P_current)
        if f_c > f_0 then
            if Depth(P_current) = m then
                (f_0, x_0) ← (f_c, x_c)
                count ← 0
                (f_1, x_1) ← Sample(P_current, 𝒫_L, ε, N_p, N)
                if f_1 > f_0 then
                    (f_0, x_0) ← (f_c, x_c)
                end if
            end if
            P_current ←NextNode(P_current)
        else
            P_current ←Backtrack(P_current)
        end if
        count ← count + 1
        if count > TB then
            if FindPruneNode(P_current) then
                P_current ←Backtrack(P_current)
            end if
            count ← 0
        end if
    end while
    if f_0 = −∞ then
        return Infeasible
    end if
    (f*_P̃, x*_P̃) ← (f_0, x_0)
    return (f*_P̃, x*_P̃)
end function
```

算法改进策略：

策略一：利用区间分析思想，减少线性规划子问题数量。检查变量上下界与约束不等式的相容性，若不相容，则把该约束不等式去掉。

策略二：采样线性规划子问题，利用非参数估计方法和遗传算法，获得尽可能大的可行解，减少访问节点数量。

策略三：出现在两层之间反复搜索时，启发式重构等价的搜索子树，增加剪枝的可能性。

实验结果与分析

- 实现求解工具RS-LPT，使用IBM CPLEX作为线性规划求解器
- 与析取规划的两种优势算法Big-M, Convex Hull（卡耐基梅隆大学）进行比较
- 与优化模理论(OMT)的三个最好的工具进行比较：Symba（多伦多大学）、OPT-MathSAT（Trento大学）、vZ（微软研究院）
- RS-LPT能够同时保证运算速度和精度（全局最优）
- RS-LPT比OMT工具的最大加速比超过100：1
- RS-LPT较Convex Hull的最大加速比超过130：1
- RS-LPT在大规模问题上较Big-M有优势。RS-LPT在所有问题上均能得到全局最优，但Big-M不能。
- 问题规模越大，RS-LPT优势越明显，可以处理超过20000个不等式和20000个逻辑符号

测试用例A：实时系统单调速率优化设计问题

$$\max \quad U = \sum_{i=1}^{n} C_i / T_i$$

$$s.t. \quad \varphi = \bigwedge_{1 \le i \le n} \bigvee_{t \in P_{i-1}(T_i)} \sum_{j=1}^{i} \lceil t / T_j \rceil C_j \le t$$

$$\psi = \bigwedge_{i=1}^{n} (C_i \ge C_i^{\min}) \wedge \bigwedge_{i=1}^{n} (C_i \le C_i^{\max})$$

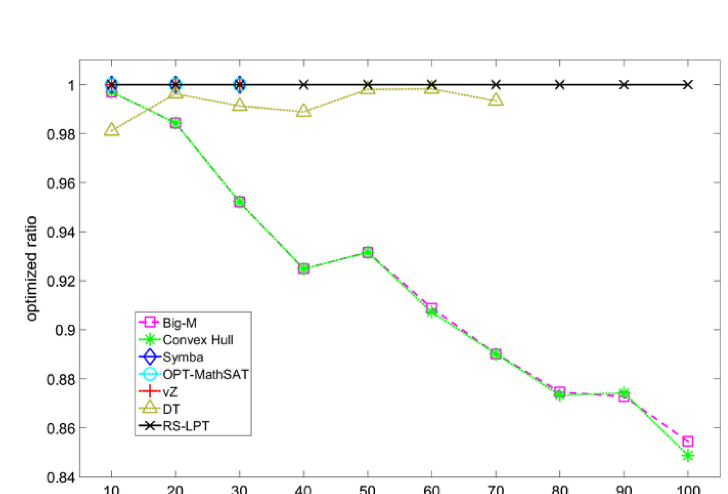
$$P_i(t) = \begin{cases} \{t\} & if\ i = 0 \\ P_{i-1}(\lfloor t/T_i \rfloor \cdot T_i) \cup P_{i-1}(t) & otherwise \end{cases}$$

测试用例B：模糊逻辑线性优化问题

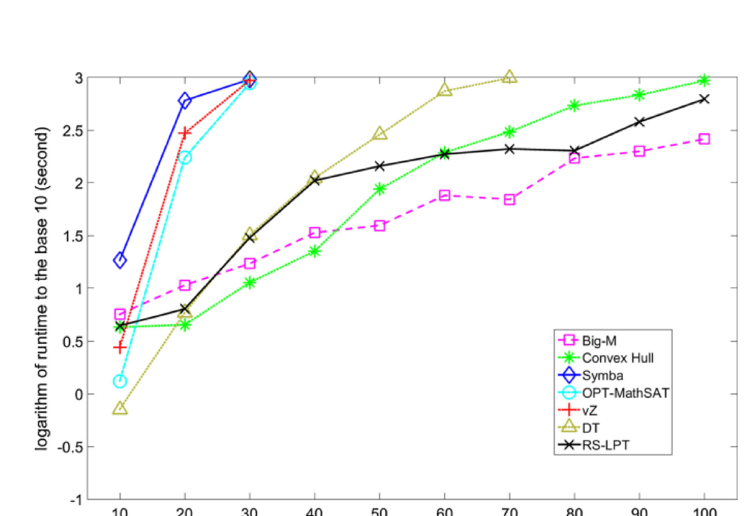
$$\max \quad f = \sum_{i=1}^{n} (c_i \sum_{j=1}^{r} p_{ij} y_j)$$

$$s.t. \quad \varphi = \bigwedge_{1 \le i \le m} \bigvee_{\substack{1 \le j \le n \\ a_{ij} \ge b_i}} \left[\sum_{k=1} p_{jk} y_k \ge b_i \right]$$

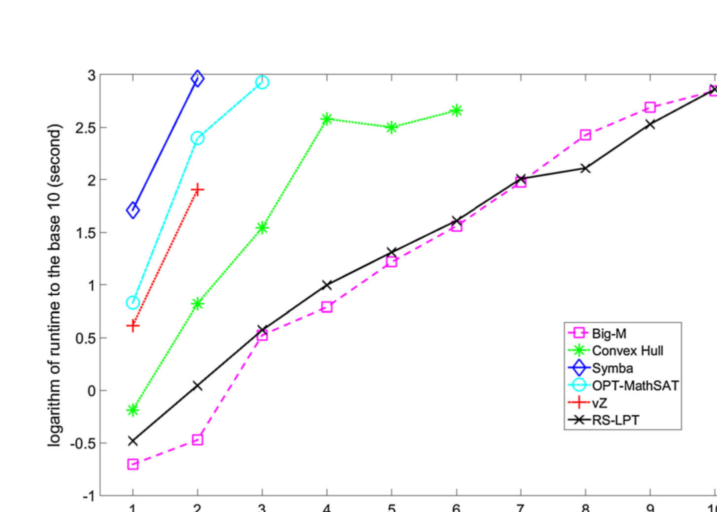
$$\psi = \bigwedge_{i=1}^{r} (y_i \ge y_i^{\min}) \wedge \bigwedge_{i=1}^{r} (y_i \le y_i^{\max})$$



各方法求解测试用例A的精度比



各方法求解测试用例A的时间



各方法求解测试用例B的时间