DisjunctiveProgramming.jl：面向 JuMP 的广义析取规划模型与算法

Hector D. Perez、Shivank Joshi、Ignacio E. Grossmann*

美国宾夕法尼亚州匹兹堡，卡内基梅隆大学
*grossmann@cmu.edu

arXiv:2304.10492v1 [cs.LO] 8 Mar 2023

摘要

我们介绍了 Julia 软件包 DisjunctiveProgramming.jl，它扩展了 JuMP.jl 的功能，使用户能够通过逻辑命题和析取约束对问题建模。随后，这些模型可重构为混合整数规划（MIP），并使用 JuMP 支持的各种 MIP 求解器求解。为此，逻辑命题会转换为合取范式（CNF），并重构为等价的代数约束。析取式则根据用户指定的重构技术（大M法或 Hull 重构）重构为混合整数约束。该软件包支持对包含线性、二次和非线性约束的析取式进行重构。

关键词

JuMP、数学优化、广义析取规划

1. 引言

对具有离散决策和连续决策的系统进行建模，通常采用代数形式的混合整数规划模型（MIP）。当问题可以仅通过线性约束和线性目标函数定义时，这类问题称为混合整数线性规划（MILP）。当可行域或目标函数中存在非线性时，这类问题称为混合整数非线性规划（MINLP）。

对这类系统进行建模的一种更系统的方法是使用广义析取规划（GDP） [5, 14], ，它将 Balas 提出的析取规划范式[2]进行了推广。GDP 支持从基于逻辑的抽象层次对系统进行建模，通过代数约束和逻辑关系捕捉支配这类系统的基本规则。这种形式化方法有助于以直观的方式表达问题，依托问题的逻辑基础，而无需引入混合整数约束。广义析取规划模型通常更易于理解，因为相关约束被分组到析取项中，用于清晰描述可行域的明确定义子集。通过 GDP 得到的模型可以重新表述为最适合相关应用的纯代数形式。此外，还常常可以利用广义析取规划模型的显式逻辑结构，相较于对应的混合整数规划模型提供更紧的松弛，从而通过先进的求解算法 [5]提高解的收敛速度和稳健性。

在优化领域，当前有大量持续进行的研究依赖广义析取规划来构建各种应用的模型。由于系统设计问题具有组合性质，广义析取规划范式已应用于复杂过程和网络 [20, 26], 能源系统的规划与最优控制 [8, 18], 以及不确定条件下化学合成的建模 [7]。这些以及其他众多广义析取规划应用表明，拥有一个稳健的广义析取规划软件包，可以大幅减少开发和测试广义析取规划模型所需的额外工作。虽然 Pyomo [6] 和 GAMS [25], 已有具备广义析取规划功能的软件包，但在 Julia 中提供此类软件包，可以大幅加快优化领域的研究；JuMP.jl [9] 等软件包正日益受到广泛关注。本文介绍了广义析取规划（GDP）范式的背景，以及对此类模型进行重构和求解的技术。随后，本文介绍了软件包 DisjunctiveProgramming.jl，它作为 JuMP.jl 的扩展，可用于创建遵循 GDP 建模范式的优化模型，并能够利用大量受支持的求解器 [9] 进行求解。一个案例研究展示了如何使用该软件包进行化工过程超结构优化。

2. 广义析取规划

GDP 建模形式是一种抽象，它同时使用代数约束和逻辑约束来描述支配系统的基本规则。将广义析取规划模型转化为等价混合整数规划模型的两种主要重构策略是大M重构 [22, 23] 和 Hull 重构 [19], ，后者能够得到更紧的模型，但代价是模型规模更大[13]。

2.1 模型

TGDP 问题的一般表示法见式（1）) - (6)。

$$\min f(x) \tag{1}$$
$$\text{s.t. } g(x) \le 0 \tag{2}$$
$$\bigvee_{i \in J_k} \begin{bmatrix} Y_{ik} \\ h_{ik}(x) \le 0 \end{bmatrix} \quad \forall k \in K \tag{3}$$
$$\Omega(Y) = true \tag{4}$$
$$Y_{ik} \in \{true, false\} \quad \forall i \in J_k, k \in K \tag{5}$$
$$x \in X \subseteq \mathbb{R}^n \tag{6}$$

此处， $f(x)$ 是要针对连续变量 x 最小化的目标函数， $g(x)$ 表示全局约束， $h(x)$ 表示特定于析取项的约束，而 Y 是控制每个析取式的布尔变量。在此记法中，共有 k 个析取式，每个析取式包含 i 个析取项。仅当相应析取项的布尔指示变量 Y_{ik} 被标记为激活状态（即 真） [5] 时，才应用约束 $h_{ik}(x) \le 0$。逻辑约束集 $\Omega(Y)$ 描述了

指示变量选择之间的逻辑关系。这些关系可以表示为逻辑命题或约束规划表达式。

在线性目标函数和约束集合的情况下，广义析取规划模型可写为式（7）至（11）。

$$\min c^T x \tag{7}$$

$$\text{s.t. } Ax \leq b \tag{8}$$

$$\bigvee_{i \in J_k} \begin{bmatrix} Y_{ik} \\ B_{ik}x \leq d_{ik} \end{bmatrix}, \quad \forall k \in K \tag{9}$$

$$\Omega(Y) = true \tag{10}$$

$$Y_{ik} \in \{true, false\} \quad \forall i \in J_k, k \in K \tag{11}$$

2.2 线性广义析取规划重构示例

线性广义析取规划系统的最简单示例见式（12）至（15），其中 Y_i 是一个布尔指示变量，当 $true$ 时，该变量强制执行析取项（$Ax \leq b$ 或 $Cx \leq d$）中的约束。

$$\begin{bmatrix} Y_1 \\ Ax \leq b \end{bmatrix} \vee \begin{bmatrix} Y_2 \\ Cx \leq d \end{bmatrix} \tag{12}$$

$$0 \leq x \leq U \tag{13}$$

$$Y_1 \veebar Y_2 \tag{14}$$

$$Y_1, Y_2 \in \{true, false\} \tag{15}$$

为便于可视化且不失一般性，使用这一简单线性示例来说明大M法和凸包重构。图1展示了一个简单线性广义析取规划的可行空间，其中包含一个析取和两个连续变量 x_1 和 x_2。左侧的矩形由左析取项 $Ax \leq b$ 中的约束描述。右侧的矩形由右析取项 $Cx \leq d$ 中的约束定义。式（14）中的异或关系保证了这两个区域不重叠。

2.2.1 大M法重构。 该问题的大M法重构由式（13）、（16）至（19）给出，其中 M 是一个足够大的标量，当未选中其指示变量时，它可以使相应约束变得冗余（即 $y_i = 0$）。请注意，布尔变量 Y_i被替换为二进制变量 y_i。当将式（19）中的整数性约束放宽为 $0 \leq x_1$时，$x_2 \leq 1$所得可行域可以通过将放宽后的模型投影到 x_1，x_2 平面上进行可视化。这将得到图2中虚线所围成的区域。需要注意的是，与原始可行解空间相比，放宽后的可行域并未达到尽可能紧的程度。较大 M值的选择决定了这种放宽的紧密程度；当模型为线性模型时，可以通过区间算术求得最优放宽对应的 M 最小值。对于非线性模型，可以通过求解最大化问题$\{\max h_{ik}(x)\colon \ x \in X\}$来获得最紧的 M。关于紧致大M法放宽的另一种方法见 [23]。

$$Ax \leq b + M \cdot (1 - y_1) \tag{16}$$

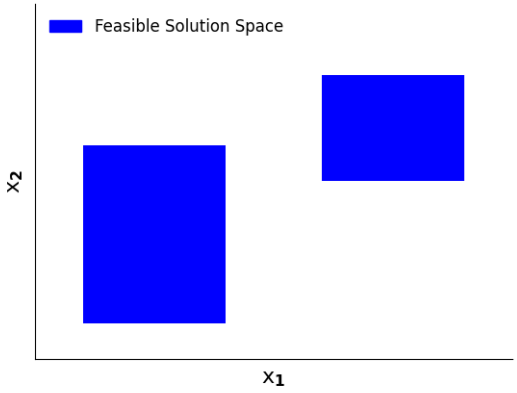


图 1. 示例析取的可行解空间

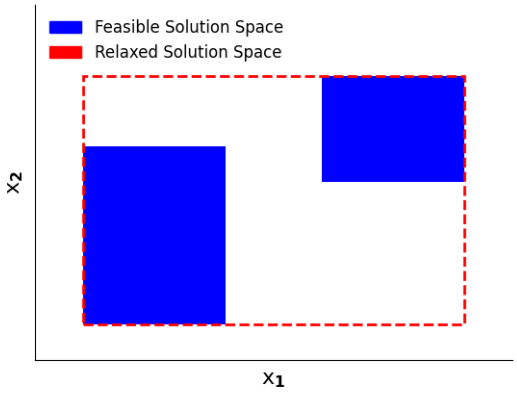


图 2. 使用大M法重构的松弛解空间

$$Cx \leq d + M \cdot (1 - y_2) \tag{17}$$

$$y_1 + y_2 = 1 \tag{18}$$

$$y_1, y_2 \in \{0, 1\} \tag{19}$$

2.2.2 Hull 重构。 Hull 重构由（13）、（18）至（23）给出，需要将模型提升到更高维空间。在投影回原始空间后，该模型的连续松弛比其大M法等价形式更紧 [14]。重构松弛可通过图 3 中虚线所包围的区域来直观表示。请注意，与图 2 中的大M重构相比，该重构提供了更紧的松弛。此外还需注意，对该松弛几何结构的描述比对大M松弛的描述更为复杂；这是由于模型中的约束和变量数量增加所致。

$$Ax_1 \leq by_1 \tag{20}$$

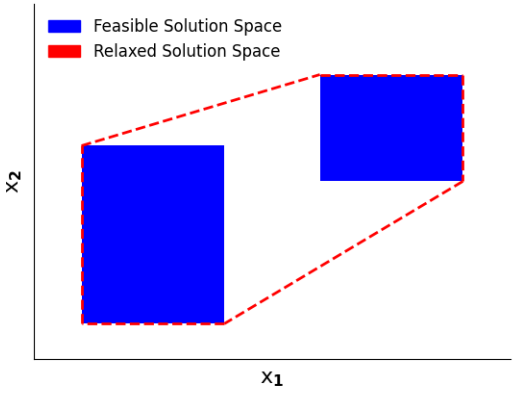


图3. 使用凸包重构的松弛解空间

$$Cx_2 \leq dy_2 \tag{21}$$

$$x = x_1 + x_2 \tag{22}$$

$$0 \leq x_i \leq Uy_i \quad \forall i \in \{1,2\} \tag{23}$$

2.3 逻辑约束重构

2.3.1 命题逻辑。 决策选择关系集合 Ω 中的逻辑命题必须转换为合取范式（CNF），才能将广义析取规划模型重构为混合整数规划模型。这意味着，命题集合中的每个子句都必须表示为析取式的合取。该过程可通过遵循命题逻辑的简化规则（德摩根定律）来完成。对于布尔变量 A、B 和 C，可使用以下规则将其转换为 CNF（按下列顺序）：

$$\begin{aligned}
A \leftrightarrow B &\text{ is replaced by } (A \rightarrow B) \wedge (B \rightarrow A)\\
A \rightarrow B &\text{ is replaced by } \neg A \vee B\\
\neg(A \vee B) &\text{ is replaced by } \neg A \wedge \neg B\\
\neg(A \wedge B) &\text{ is replaced by } \neg A \vee \neg B\\
(A \wedge B) \vee C &\text{ is replaced by } (A \vee C) \wedge (B \vee C)
\end{aligned}$$

将逻辑命题转换为合取范式后，每个子句都可以根据以下等价关系转换为代数约束（注意：在重构中，任何取反的布尔变量 $\neg Y_i$ 都会被替换为 $1 - y_i$），

$$\bigvee_{i \in I} Y_i \text{ becomes } \sum_{i \in I} y_i \geq 1$$

将命题逻辑语句转换为合取范式（CNF）还有其他方法，这些方法保留子句可满足性，而非子句等价。它们可以避免子句规模呈指数增长，并得到逻辑上自洽的结果 [17]。

2.3.2 约束规划。 也可以在 Ω(Y) 中加入类似约束规划（CP）中的选择约束。这些约束的形式为“允许布尔变量列表中恰好有 n 个元素为 $true$。”这类约束克服了布尔异或（∨）运算符的局限，因为该运算符只能强制布尔值列表中有奇数个元素为 $true$。将“恰好”替换为“至多”或“至少”，还可以得到其他类似 CP 的约束。这些约束可重述如下，

$$\text{exactly}(n, Y) \text{ becomes } n = \sum_i Y_i$$

$$\text{atleast}(n, Y) \text{ becomes } n \leq \sum_i Y_i$$

$$\text{atmost}(n, Y) \text{ becomes } n \geq \sum_i Y_i$$

如式（14）所示的异或约束，通常可建模为 $exactly(1,\ \{Y_1, Y_2\})$。

2.4 求解技术

2.4.1 析取分支定界。 析取分支定界方法与混合整数规划问题 [13] 的标准分支定界方法非常相似。首先，通过求解原始广义析取规划的大M法或 Hull 重构的连续松弛问题，初始化搜索树，以获得最优解的下界。随后，对指示二进制变量最接近 1 的析取进行分支。此时会创建两个节点：一个节点将相应的指示布尔变量固定为 $true$（强制执行该析取项），另一个节点将其固定为 $false$（从析取中移除该析取项）。对每个节点进行重构并求解，以获得候选下界。如果某个节点的解是满足所有整数性约束的可行解，则该解是最优解的上界。任何超过上界的非整数解都会从搜索树中剪枝。重复此过程，直到上下界处于所需容差范围内。

2.4.2 基于逻辑的外逼近。 基于逻辑的外逼近是另一种算法，它仿照了求解混合整数非线性规划问题的标准技术 [10]。该方法首先通过固定不同析取式中的布尔变量，确定一组约简后的非线性规划（NLP）子问题，使得每个析取项至少在这组子问题中被选择一次（集合覆盖步骤）。求解每个子问题以获得一个上界和一个可行点，并在该点处对原始广义析取规划的目标函数和约束进行线性化，然后求解所得问题（通过直接重构为混合整数线性规划（MILP），或通过析取分支定界）以找到一个下界。如果下界和上界的解尚未收敛，则固定前一个解中的布尔变量，并求解所得的 NLP，以找到一个可能更紧的上界解。重复该过程，直到收敛。

2.4.3 混合切平面。 切平面法是一种算法，用于在求解问题之前，通过添加额外约束来收紧经大M法重构的问题的松弛解空间；这些约束会移除与实际可行解空间不相交的松弛区域。这些针对松弛解空间的“切平面”源自更紧的凸包松弛，即

问题的凸包松弛。该算法在 Hull 重构的复杂性及相应计算开销，与较不紧的 Big-M 重构之间提供了一种折中。[24]。

3. DisjunctiveProgramming.jl

以下部分介绍了 DisjunctiveProgramming.jl 软件包的功能，并通过一个来自化学加工行业、用于超结构优化的示例展示其语法。此外，还展示了嵌套析取的用法。

3.1 特性

DisjunctiveProgramming.jl 允许通过 @disjunction 宏或 add_disjunction! 函数定义包含析取式的 JuMP 模型，并直接通过大M法或 Hull 方法进行重构。大M值可以针对整个析取、每个析取项，或每个析取项中的每个约束分别指定。另外，如果约束是线性的，代码可以利用变量界限对每个约束执行区间运算，以确定要使用的最紧 M 值 [1]。对于非线性 GDP 约束，Hull 重构中透视函数采用 epsilon 近似公式 [11]。用户可以指定要使用的 epsilon 容差值。透视函数通过 Symbolics.jl [12] 对符号表达式进行操作来生成。

可以在 JuMP 模型中使用析取指示变量和标准布尔运算符（$\Leftrightarrow$、$\Rightarrow$、$\vee$、$\wedge$、以及 $\neg$）构成表达式，并通过 @proposition 宏添加逻辑命题。这些命题会自动转换为合取范式，并作为整数代数约束添加到模型中。也可以使用 choose! 函数添加约束规划约束。这些表达式同样会自动重构为整数代数约束。

也支持嵌套析取。

3.2 示例

为说明 DisjunctiveProgramming.jl（版本 0.3.3）的语法，请考虑图 4 所示化工过程的简单超结构优化问题。该问题需要设计一座采用两种候选反应器技术（R_1 和 R_2）之一的化工厂。如果选择第二种反应器技术，还必须安装分离系统，且有两种可选的分离技术（S_1 和 S_2）。该广义析取规划模型旨在最大化产品流量（F_7），同时将反应器（C_R）和分离器（C_S）的安装成本计入折减，如式 (24) 所示；模型还需满足式 (25) 中的嵌套析取以及式 (26) - (27) 中的全局质量平衡。系统变量包括各流股上的流量 i（F_i）和安装成本，其相应的上下界见式 (28) - (30)。固定成本和工艺收率参数分别为 γ 和 β。

$$\max F_7 - C_R - C_S \tag{24}$$

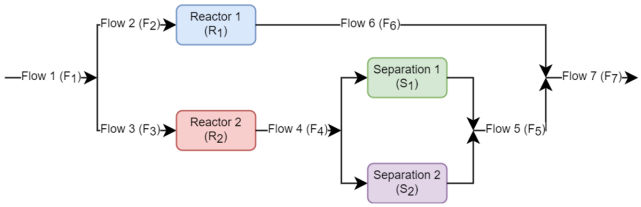


图 4. 示意性超结构优化问题

$$\begin{bmatrix} Y_{R1} \\ F_6 = \beta_{R1} F_2 \\ F_3 = 0 \\ F_4 = 0 \\ F_5 = 0 \\ C_R = \gamma_{R1} \\ CS = 0 \end{bmatrix} \vee \begin{bmatrix} Y_{R2} \\ F_2 = 0 \\ F_6 = 0 \\ F_4 = \beta_{R2} F_3 \\ C_R = \gamma_{R2} \\ \begin{bmatrix} Y_{S1} \\ F5 = \beta_{S1} F_4 \\ C_S = \gamma_{S1} \end{bmatrix} \vee \begin{bmatrix} Y_{S2} \\ F5 = \beta_{S2} F_4 \\ C_S = \gamma_{S2} \end{bmatrix} \end{bmatrix} \tag{25}$$

$$F_1 = F_2 + F_3 \tag{26}$$

$$F_7 = F_5 + F_6 \tag{27}$$

$$0 \le F_i \le 10 \quad \forall i \in \{1, ..., 7\} \tag{28}$$

$$0 \le C_S \le C_S^{max} \tag{29}$$

$$C_R^{min} \le C_R \le C_R^{max} \tag{30}$$

上述系统可使用 DisjunctiveProgramming.jl 建模，并通过大M重构进行重构。随后，使用 HiGHS 开源混合整数线性规划求解器对得到的 JuMP 模型进行优化，如下所示。 [16]

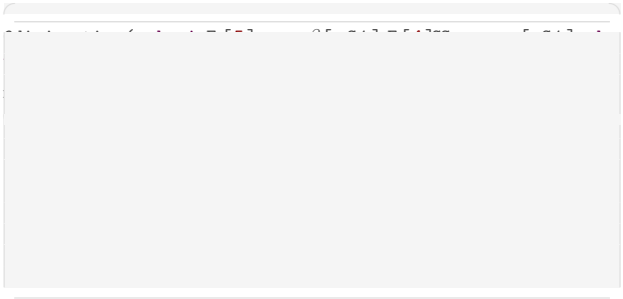
(1) 创建 JuMP 模型，并定义模型变量和全局约束（质量平衡）。

```
使用 析取规划 ,                                    JuMP,    HiGHS

# 创建模型
m   = JuMP . 模型 ( HiGHS . 优化器 )
# 向模型添加变量
@variable ( m,    0  <= F [i  =  1: 7]  <=  10)
@变量 ( m       ,  0  <= CS   <= CSmax )
<escape>@variable ( m, CRmin   <= CR   <= CRmax )</escape>

# 向模型添加约束
@constraints ( m ,
      开始
            F [1]  == F [2]  + F [3]
            F [7]  == F [5]  + F [6]
      end
)
```

（2）使用 @disjunction 宏，为超结构中的分离技术定义内部（嵌套）析取。



(3) 在外层析取中定义约束。

(4) 使用（3）中定义的约束块和 add_disjunction! 函数构建主析取。请注意，嵌套析取的重新表述后的约束存储在模型的 .ext 字典中，键名为该析取的名称（本例中为 :YSin）。

(5) 使用 choose! 函数添加选择逻辑约束。第一个约束确保只选择一个反应器（即 $Y_{R_1} \vee Y_{R_2}$）。第二个约束确保仅当选择第二个反应器（R_2）时才定义分离系统。该约束等价于命题 $Y_{R_2} \Leftrightarrow Y_{S_1} \vee Y_{S_2}$。

```
选择!(1,YR[1],YR[2]; 模式  =: 恰好)选择!(YR[2],
```

（6）添加目标函数并进行优化。

```
@objective(m,Max,F[7]-CS-CR)优化!
```

4. 未来工作

DisjunctivePrograming.jl 软件包的后续开发将进一步扩展 JuMP.jl，使其能够创建在模型创建时不一定需要进行重构的广义析取规划模型。此类模型将支持使用不同的广义析取规划求解策略，例如直接重构为混合整数（非）线性规划、析取分支定界、基于逻辑的外逼近、混合切平面和基本步骤。更新后的软件包将利用 JuMP 现有的扩展基础设施，并支持为析取定义索引记法、新的布尔变量类型以及新的析取约束类型。预计这些改进将使该软件包在 JuMP 中用于广义析取规划的高级应用时更加易用、灵活且高效。

5. 相关工作

广受欢迎的 Python 软件包 Pyomo [4, 15] 广泛用于优化开发，并包含广义析取规划扩展 [6]。GAMS [3] 是一种广泛使用的优化建模语言，通过采用 LogMIP [25]的 GAMS EMP 求解器支持 GDP。当前也有研究致力于将 Aspen 等现代过程模拟技术集成到 GDP 范式中 [21]。

6. 结论

DisjunctiveProgramming.jl 是 JuMP 的扩展，用于创建按照广义析取规划范式构建的优化模型。该软件包提供了多种重构选项，包括大M法和凸包松弛。利用 JuMP 中现有的数学规划基础设施，可以使用该软件包对问题进行建模、重构和优化。这对于 GDP 的工业和学术应用十分有用，例如超结构优化。该软件包的功能使这一建模范式能够借助 Julia 高效的动态类型系统进行快速开发、构建和测试，从而实现优化模型。

7. 参考文献

[1] Ashish Agarwal、Sooraj Bhat、Alexander Gray 和 I gnacio E Grossmann。自动化数学规划转换。载于国际声明式语言实用方面研讨会，第134–148页。Springer，2010年。doi:10.1007/978-3-642-11503-5 12.._[2] Egon Balas。析取规划。Springer，2018年。[3] Michael R. Bussieck 和 Alex Meeraus。通用代数建模系统（*GAMS*），第137–157页。Springer US，波士顿，马萨诸塞州，2004年。doi:10.1007/978-1-4613-0215-5_8. [4] Michael L. Bynum、Gabriel A. Hackebeil、William E. Hart、Carl D. Laird、Bethany L. Nicholson、John D. Siirola、Jean-Paul Watson 和 David L. Woodruff。*Pyomo——Python*中的优化建模，第67卷。Springer Science & Business Media，第三版，2021年。[5] Qi Chen 和 Ignacio Grossmann。使用广义析取规划的现代建模范式。过程，7(11)，2019年。doi:10.3390/pr7110839。[6] Qi Chen、Emma S Johnson、David E Bernal、Romeo Valentin、Sunjeev Kale、Johnny Bates、John D Siirola 和 Ignacio E

Grossmann。Pyomo.gdp：用于基于逻辑的建模与优化开发的生态系统。《优化与工程》，23(1):607–642，2022年。[7] Ying Chen、Yixin Ye、Zhihong Yuan、Ignacio E. Grossmann 和 Bingzhen Chen。将随机规划与可靠性整合到化学过程的最优综合中。《计算机与化学工程》，157 107616，2022 10 1016 2021 107616 : . doi: . /j.compchemeng. . .[8] Seolhee Cho 和 Ignacio E. Grossmann。可靠发电系统扩展规划的优化模型。载于Ludovic Montastruc 和 Stephane Negny（编），第*32*届欧洲计算机辅助过程工程研讨会，第51卷，计算机辅助化学工程，第841–846页。Elsevier，2022年。doi:10.1016/B978-0-323-95879-0.50141-7。[9] Iain Dunning、Joey Huchette 和 Miles Lubin。JuMP：用于数学优化的建模语言。《*SIAM*评论》，59(2):295–320，2017年。doi:10.1137/15M1020575。[10] Ignacio E. Grossmann。基于逻辑的外逼近，第1928–1931页。Springer US，波士顿，马萨诸塞州，2009年。doi:10.1007/978-0-387-74759-0 348。_[11] Kevin C. Furman、Nicolas W. Sawaya 和 Ignacio E. Grossmann。使用透视函数表示非线性析取凸集的一种计算上实用的代数表示。《计算优化与应用》，76(2):589–614，2020年。doi:10.1007/s10589-020-00176-0。[12] Shashi Gowda、Yingbo Ma、Alessandro Cheli、Maja Gwó´zzd´z、Viral B. Shah、Alan Edelman 和 Christopher Rackauckas。通过多重分派实现高性能符号数值计算。*ACM*计算代数通讯，55(3):92–96，2022年1月。doi:10.1145/3511528.3511535。[13] Ignacio E. Grossmann 和 Sangbum Lee。广义凸析取规划：非线性凸包松弛。《计算优化与应用》，26 100，2003 1 83 10 1023 1025154322278 ( ): – . doi: . /a: .[14] Ignacio E. Grossmann 和 Francisco Trespalacios。通过广义析取规划对离散-连续优化模型进行系统建模。《*AIChE*期刊》，59(9):3276–3295，2013年。doi:10.1002/aic.14088。[15] William E Hart、Jean-Paul Watson 和 David L Woodruff。Pyomo：使用Python对数学规划进行建模和求解。《数学规划计算》，3(3):219–260，2011年。doi:10.1007/s12532-011-0026-8。[16] Qi Huangfu 和 JA Julian Hall。并行化对偶修正单纯形法。《数学规划计算》，10(1):119–142，2018年。doi:10.1007/s12532-017-0130-5。[17] Paul Jackson 和 Daniel Sheridan。布尔电路的子句形式转换。《可满足性测试理论与应用》*style>*，第*183–198*页，*2005*年。*doi:10.1007/.Donghun Kim*。基于广义析取规划的混合整数线性模型预测控制表述，用于区域能源系统的光伏自发自用与电网脱碳的优化运行：现场实施。国际高性能建筑会议，2022年。[19] Sangbum Lee 和 Ignacio E. Grossmann。非线性广义析取规划的新算法。《计算机与化学工程》，24(9):2125–2141，2000年。doi:10.1016/S0098-1354(00)00581-0。[20]Fahad Matovu、Shuhaimi Mahadzir、Rasel Ahmed 和 Nor Erniza Mohammad Rozali。综合与采用广义析取规划优化多级制冷系统。《计算机与化学工程》，163:107856，2022年。doi:10.1016/j.compchemeng.2022.107856.[21]Miguel A. Navarro-Amorós、Rubén Ruiz-Femenia 和 José A. Caballero。在广义析取规划框架下集成模块化过程模拟器，以实现结构化流程图优化。《计算机与化学工程》，67:13–25，2014年。doi:10.1016/j.compchemeng.2014.03.014.[22] George L. Nemhauser。整数与组合优化。约翰·威立父子出版公司，1999年。[23]Francisco Trespalacios 和 Ignacio E. Grossmann。针对广义析取规划的改进大M重构。《计算机与化学工程》，76 103，2015 98 10 1016 2015 02 013 : – 。doi: . /j.compchemeng. . . .[24] Francisco Trespalacios 和 Ignacio E. Grossmann。凸广义析取规划的割平面算法。《*INFORMS*计算学报》，28(2):209–222，2016年。doi:10.1287/ijoc.2015.0669.[25] Aldo Vecchietti 和 Ignacio E Grossmann。LogMIP：面向过程系统模型的析取式0–1非线性优化器。《计算机与化学工程》，23(4-5):555–565，1999年。doi:10.1016/s0098-1354(98)00293-2.[26] Wenjin Zhou、Kashif Iqbal、Xiaogang Sun、Dinghui Gan、Chun Deng、José María Ponce-Ortega 和 Chunmao Chen。采用析取规划模型合成基于性质的多种资源供水网络。《化学工程研究与设计》，187:69–83，2022年。doi:10.1016/j.cherd.2022.08.027。