# Gurobi 求解器日志解析

2025 年 12 月 18 日

# 整体结构

1. **第 1 章：Header 信息**
   求解环境、参数设定、模型规模、系数统计
2. 第 2 章：Presolve 与模型简化
   预处理行为、规模收缩、早期不可行/无界检测
3. 第 3 章：根节点与 B&B 搜索结构
   根 LP/Barrier、节点日志、Gap 收敛
4. 第 4 章：总结块与二次分析
   总结行、性能画像、自动化调参与监控

# Section 1 : Header

## Gurobi Header 核心内容

- 求解器版本、平台、CPU 信息、线程数
- Non-default parameters（非默认参数表）
- 模型规模：Rows / Columns / Nonzeros
- 变量类型统计、Model fingerprint
- Coefficient statistics（矩阵、目标、边界、RHS 范围）

# Header：求解器版本与平台

```
Gurobi Optimizer version 12.0.3 build v12.0.3rc0
  (linux64 - "Ubuntu 24.04.2 LTS")
```

- **版本号与 build**：
  - 主版本、次版本、修订版本以及构建标识
  - 与算法实现、默认参数、数值稳定性相关
- **平台信息**：
  - 操作系统与架构（如 `linux64`）
  - 不同平台可能使用不同 BLAS / LAPACK 库
- **分析用途**：
  - 实验可复现性：不同版本的日志不宜直接横向对比
  - 部署迁移时，检查性能变化是否来自版本/平台

# Header：CPU 结构与线程数

```
CPU model: AMD Ryzen Threadripper 7970X 32-Cores,
           instruction set [SSE2|AVX|AVX2|AVX512]
Thread count: 32 physical cores, 64 logical processors,
              using up to 32 threads
```

- **CPU 模型与 SIMD 指令集**：
  - 影响线性代数内核与 Barrier 性能
- **线程配置**：
  - Threads $= 32$ 为本次求解的最大并行度
  - 对 MIP：可并行探索多个节点/子树
  - 对 LP/Barrier：可并行分解与矩阵运算

# Header: Classes of Non-default parameters

| 参数类别 | 属性说明 | 典型参数示例 |
| --- | --- | --- |
| Termination | 停止参数，用于控制求解过程的终止条件 | `TimeLimit`：设置最大求解时间；<br>`SolutionLimit`：限制返回的 MIP 可行解数量 |
| Tolerances | 容差参数，用于控制最优性与可行性精度 | `MIPGap`：设定 MIP 的相对最优性间隙；<br>`FeasibilityTol`：设置约束可行性容差 |
| Simplex | 单纯形法相关参数，控制 LP 求解行为 | `InfUnbdInfo`：控制是否返回<br>不可行或无界模型的额外诊断信息 |
| Barrier | 障碍法相关参数，控制内点法求解过程 | `QCPDual`：控制是否<br>计算并返回二次约束模型的对偶值 |
| MIP | 混合整数规划参数，控制分支定界与搜索策略 | `BranchDir`：设定优先分支方向；<br>`Heuristics`：设置启发式算法<br>在总时间中所占的比重 |
| MIP Cuts | 割平面参数，用于控制割平面生成强度 | `Cuts`：设定割平面算法的总体强度 |
| Tuning | 调参工具相关参数，用于自动参数优化 | `TuneCriterion`：设定调参目标准则；<br>`TuneTimeLimit`：设定调参的时间上限 |
| Multiple Solutions | 多解搜索参数，用于存储和返回多个可行解 | `PoolSolutions`：控制可行解池中<br>存储的解数量 |

# Header: 常用 paras

| 常用参数 | 属性说明 | 取值与含义 |
| --- | --- | --- |
| TimeLimit | 时间限制参数 | 单位：秒；达到给定时间后终止求解并返回当前结果 |
| MIPFocus | 设定 MIP 的求解侧重点 | 0：默认，均衡搜索可行解与最优性证明；<br>1：侧重快速找到可行解；<br>2：侧重证明最优性；<br>3：侧重界的提升（适用于界提升缓慢的情形） |
| Method | 设定线性模型或 MIP 根节点的求解方法 | -1：默认（LP 使用 Method=3；QP/QCP 使用 Method=2；MIP 根节点自动选择）；<br>0：原始单纯形法；<br>1：对偶单纯形法；<br>2：内点法（Barrier）；<br>3：并发执行 0/1/2（结果可能因运行而异）；<br>4：确定性并发 0/1/2（重复运行结果一致）；<br>5：确定性并发 0/1 |
| LogToConsole | 控制日志是否输出到控制台 | 0：关闭控制台输出；<br>1：开启控制台输出（默认） |
| LogFile | 设置日志文件名称 | 字符串类型；默认空字符串（不输出到文件） |
| Presolve | 控制预处理强度 | -1：默认（自动选择）；<br>0：关闭预处理；<br>1：保守预处理；<br>2：进取预处理（耗时较多，但模型更紧凑） |
| MIPGap | 相对最优性间隙阈值 | 默认 1e-4；当 $\frac{\lvert \mathrm{UB}-\mathrm{LB} \rvert}{\max(1,\lvert \mathrm{UB} \rvert)} \leq$ MIPGap 时终止计算 |

# Header：参数信息的分析价值

- **实验设定的完整记录**：
  - 所有非默认参数构成一次求解的“策略配置”
  - 算法对比、消融实验需严格对齐此部分
- **求解模式推断**：
  - `MIPGap = 0` 表示要求严格最优证明
  - 若同时限定 `TimeLimit`，则常出现“时间到而非 Gap 收敛”的情形
- **自动化工具输入**：
  - 日志解析器可以读取并存储 Non-default parameters
  - 支持后续的自动调参与策略回顾

# Header：模型规模

```
Optimize a model with 4381696 rows,
                      1101446 columns
                      15327041 nonzeros
Model fingerprint: 0xa13f87a3
Variable types: 3142 continuous,
                1098304 integer (1098304 binary)
```

- **问题规模**：m（约束）/n（变量）/nnz（非零元素）
- **变量类型**：continuous/integer/binary

# 问题规模对求解的影响（一）：约束数 $m$

**1. 数学层面**

在线性规划（LP）与混合整数规划（MIP）中，约束数 $m$ 决定了：

- 可行域的维度约束结构；
- 单纯形法中基矩阵 $B \in \mathbb{R}^{m \times m}$ 的规模；
- 内点法（Barrier）中 KKT 系统的维度。

**2. 对 LP 求解复杂度的直接影响**

- 单纯形法：每一次 pivot 需要解

$$Bx = b, \quad B \in \mathbb{R}^{m \times m}$$

- Barrier 法：每次迭代需解

$$(AD^2A^\top)\Delta y = r, \quad A \in \mathbb{R}^{m \times n}$$

因此：

$$m \uparrow \Rightarrow \text{线性系统规模增大} \Rightarrow \text{单次 LP 计算成本上升}$$

# 问题规模对求解的影响（二）：变量数 $n$ 与非零元 $\mathrm{nnz}$

**2. 非零元数量 $\mathrm{nnz}$ 的核心地位**

- $\mathrm{nnz}(A)$ 决定矩阵存储与访存成本；
- 决定稀疏 LU / Cholesky 分解的填充（fill-in）；
- 直接影响每一次线性代数运算复杂度。

**3. 算法复杂度中的体现**

- 单纯形法：

$$T_{\text{simplex}} \approx (\#\text{pivots}) \times \mathcal{O}(\mathrm{nnz})$$

- Barrier 法：

$$T_{\text{barrier}} \approx (\#\text{iter}) \times \mathcal{O}(\mathrm{nnz}^{1.5})$$

**结论**：

$\mathrm{nnz}$ 是 LP/MIP 求解中最核心的“隐藏规模指标”

# 变量类型对求解的影响（一）：连续变量与整数变量

**1. 连续变量（Continuous Variables）**

- 仅影响 LP 的线性代数规模；
- 不引入离散搜索；
- 对应的计算复杂度主要由 $m, n, \mathrm{nnz}$ 决定。

**2. 整数变量（General Integer Variables）**

- 引入分支定界（Branch-and-Bound）搜索；
- 每个整数变量理论上引入一个二叉分支；
- 搜索树规模在最坏情况下指数增长：

**3. Root LP 的关键作用**

- 整数变量在 root LP 中被松弛为连续；
- LP 解的分数结构决定分支变量候选；
- 决定初始下界、分支顺序与伪成本初始化。

**结论**：

$$\text{整数变量数} \Rightarrow \text{搜索复杂度的主要来源}$$

# 变量类型对求解的影响（二）：二进制变量与退化

**1. 二进制变量的几何特性**

- Root LP 中线性目标函数倾向将变量推向端点 $0/1$；
- 大量界约束在最优解处成为紧约束。

**2. 退化的产生机制**

- 紧约束数量 $\gg n$；出现多个等价基；单纯形法中发生零步长 pivot。

**3. 对求解性能的影响**

- Root LP simplex iterations 激增；
- 基矩阵更容易 ill-conditioned；
- 数值误差积累，触发 numerical trouble。

**4. 在 MIP 中的连锁效应**

- Root LP 变慢；
- 伪成本估计噪声增大；
- 分支效率下降，搜索树变大。

# Header：Coefficient statistics

```
Coefficient statistics:
  Matrix range     [1e-03, 2e+03]
  Objective range  [1e+00, 1e+04]
  Bounds range     [1e+00, 2e+03]
  RHS range        [1e+00, 5e+02]
```

- **Matrix range**：约束矩阵 $A$ 中非零系数的绝对值范围。容易导致数值不稳定，出现 Numerical trouble 或迭代异常多。
- **Objective range**：目标函数系数范围。导致某些变量“主导”优化，不利于 Gap 的数值稳定。
- **Bounds range**：变量上下界的数值范围
- **RHS range**：约束右端项 $b$ 的数值范围，大 M 直接让根节点 Barrier 很难收敛。

# 从 Header 的 Coefficient statistics 到数值难点

- Header 报告的 `Matrix/Object/Bounds/RHS range` 是对模型数值尺度的粗粒度刻画。
- 经验上，大范围系数常伴随线性系统 *ill-conditioning*，使浮点误差被显著放大。

## 关键链路（后续推导）

大范围系数 $\Rightarrow$ $\kappa(\cdot)$ 变大 $\Rightarrow$ 线性系统解误差放大 $\Rightarrow$ LP/Barrier 变慢与数值修正 $\Rightarrow$ MIP 下界噪声 $\Rightarrow$ Ga

# 线性系统扰动理论：条件数如何放大误差

考虑线性系统

$$Ax = b, \quad A \in \mathbb{R}^{n\times n}.$$

实际计算等价于解一个被扰动的系统：

$$(A + \Delta A)\hat{x} = b + \Delta b.$$

经典扰动界（忽略高阶项）：

$$\frac{\|\hat{x} - x\|}{\|x\|} \lesssim \kappa(A)\left(\frac{\|\Delta A\|}{\|A\|} + \frac{\|\Delta b\|}{\|b\|}\right), \qquad \kappa(A) = \|A\| \cdot \|A^{-1}\|.$$

- 双精度机器精度 $u \approx 10^{-16}$，浮点误差通常满足 $\|\Delta A\|/\|A\| = \mathcal{O}(u)$。
- 因此 $\kappa(A)$ 大时，$\mathcal{O}(u)$ 级扰动可被放大到可观误差。

# 系数范围与病态性：从 range 到 $\kappa$ 的直觉连接

Header 的 `Matrix range` 给出

$$a_{\min} = \min_{a_{ij}\neq 0} |a_{ij}|, \quad a_{\max} = \max_{a_{ij}\neq 0} |a_{ij}|.$$

定义数量级跨度：

$$\Delta_A = \log_{10}\left(\frac{a_{\max}}{a_{\min}}\right).$$

- $\Delta_A$ 很大（例如 $10 \sim 15$）时，常出现：
  - 列/行尺度极不均衡，导致几何上“近似共线/近似平行”的约束增多；
  - 基矩阵或牛顿系统更易接近奇异（ill-conditioned）。
- 同理，对 `Bounds range`、`RHS range` 的巨大跨度，往往对应 big-M 或单位尺度不一致，进一步恶化数值稳定性。

## 结论（面向日志）

$\Delta$ 大 $\Rightarrow$ $\kappa$ 可能很大 $\Rightarrow$ 线性系统解误差上升，触发 `scaling/perturbation/numerical trouble`。

# 为什么根节点 LP（Simplex）会变慢：基矩阵误差与退化

单纯形法每一步需要解基矩阵系统：

$$B\,\Delta x_B = r,$$

其中 $B$ 为约束矩阵的某个基子矩阵。若 $\kappa(B)$ 大，则

$$\frac{\|\widehat{\Delta x}_B - \Delta x_B\|}{\|\Delta x_B\|} \lesssim \kappa(B) \cdot \mathcal{O}(u).$$

- **方向误差**：步长方向不稳，导致 pivot 选择与迭代行为更不可靠。
- **退化放大**：当模型高度退化（许多约束“几乎 binding”）时，小误差即可造成：
  - 大量近零步长（迭代“空转”）；
  - 更多迭代与更频繁的重因子分解（factorization）。

## 日志对应

根节点 LP/dual simplex 迭代数异常大；出现 `Basis ill-conditioned`、`numerical trouble`、更长的 root 时间。

# 为什么根节点 Barrier 会变慢：牛顿系统条件数恶化

Barrier/内点法每步需要解牛顿系统（示意）：

$$(AD^2A^\top)\,\Delta y = \text{rhs},$$

其中 $D$ 为依赖当前迭代点的对角矩阵。粗略有：

$$\kappa(AD^2A^\top) \lesssim \kappa(A)^2 \cdot \kappa(D^2).$$

- 若 $A$ 已病态（$\kappa(A)$ 大），再叠加 barrier 迭代中 $D$ 的巨大尺度变化，牛顿系统更难稳定求解。
- 结果表现为：更多迭代、更多线性系统求解开销、更多数值修正（scaling / refinement / 线搜索回退）。

## 日志对应

`scaling applied in ... passes`、Barrier 迭代耗时显著、root 耗时占比上升。

# 为何 MIP 下界不稳、Gap 收敛慢：LP bound 的噪声水平

在 B&B 中，每个节点解 LP 松弛得到节点下界 $\hat{z}$。当数值误差显著时，可将

$$\hat{z} = z^\star + \epsilon_z, \qquad \epsilon_z \approx \mathcal{O}(\kappa(\cdot)\, u) \times (\text{问题尺度}).$$

全局上下界：

$$\mathrm{LB} = \max(\text{node bounds}), \qquad \mathrm{UB} = \min(\text{incumbents}),$$

Gap：

$$\mathrm{Gap} = \frac{\mathrm{UB} - \mathrm{LB}}{\max(1, |\mathrm{UB}|)}.$$

- 当 $|\epsilon_z|$ 接近目标的精度要求（如 $\texttt{MIPGap} = 10^{-4}$ 量级）时，
  - 节点 bound 判断与剪枝变得“噪声驱动”；
  - LB 曲线出现抖动/平台，Gap 难以平滑收敛。

## 日志对应

`BestBd` 增长缓慢或抖动、Gap 长时间平台；更多节点被保留与扩展（剪枝效率下降）。

# 为何 runtime 对微小扰动高度敏感：连续敏感性 + 离散决策放大

当 $\kappa$ 很大时，LP 解与 bound 对微小扰动高度敏感：

$$\frac{\|\hat{x} - x\|}{\|x\|} \lesssim \kappa(\cdot)\,\varepsilon.$$

而 B&B 中存在大量阈值型离散决策（示意）：

$$\text{if } f(\hat{x}, \hat{z}) > \alpha \Rightarrow \text{选择 A；否则选择 B.}$$

因此微小扰动可能翻转分支变量选择、节点选择与剪枝判断，导致搜索树路径显著不同。

## 指数型放大（直观）

在大规模二进制变量下，节点规模可呈指数增长：

$$\#\text{nodes} \sim c^{n_{\text{bin}}}, \quad c > 1,$$

因此“路径差异”可带来若干数量级的 runtime 差异。

- 体现为：同一实例在不同平台/线程/微小数据扰动下，节点数与时间剧烈波动。

# 可操作结论：日志信号 ⇒ 数学原因 ⇒ 应对方向

| 日志信号 | 数学原因（核心） | 优先应对方向 |
|---|---|---|
| range 跨度巨大 | $\kappa$ 大，误差放大 | 单位缩放；tight bounds；减小 big-M |
| root LP/Barrier 很慢 | 线性系统病态，迭代/修正增多 | 改模型尺度；必要时提高 NumericFocus |
| BestBd 抖动/Gap 平台 | LP bound 噪声 $\epsilon_z$ 不可忽略 | 调整精度目标；改模型数值质量 |
| runtime 波动巨大 | 连续敏感性 + 离散决策翻转 | 实例分层；数值风险打标与策略分流 |

## 核心原则

参数可“止血”（稳定性优先），但数值问题的根治通常来自建模：尺度一致、界更紧、big-M 更小。

# Header 信息综合视图

| 维度 | 典型字段 | 主要作用 |
|---|---|---|
| 环境 | 版本、平台、CPU、Threads | 可复现性、并行能力、硬件依赖 |
| 参数 | Non-default parameters | 求解策略与终止条件的完整描述 |
| 规模 | Rows, Columns, Nonzeros, Var types | 估计 LP/MIP 难度与存储需求 |
| 结构 | Model fingerprint | 模型版本标识，防止结构漂移 |
| 数值 | Coefficient statistics | 预判数值稳定性与 big-M 风险 |

- Header **提供了一个静态视角**：在求解开始前即刻，关于“问题 + 环境 + 策略”的完整快照
- 后续章节的 Presolve、Root、节点日志可视为对这一快照的**动态响应**

# Section 2 : Presolve

1. Presolve 在日志中的位置与流程
2. 规模收缩：Rows/Cols/Nonzeros 的变化
3. 结构与逻辑简化：聚合、替代、固定、界收紧
4. 早期判定：infeasible / unbounded / trivial
5. Presolve 时间与数值信息
6. 可提炼指标与二次开发价值

# Presolve：目标、原则与数学本质

**Presolve 的核心目标**

- 在不改变可行域与最优解的前提下，简化模型；
- 降低问题规模（$m, n, \mathrm{nnz}$）；
- 减少退化与数值不稳定风险。

**数学基本原则**

Presolve 中的所有操作均满足：

$$\mathcal{F}_{\text{original}} = \mathcal{F}_{\text{presolved}}, \quad z^{\star}_{\text{original}} = z^{\star}_{\text{presolved}}$$

**三类核心作用**

- **降维**：消除冗余变量与约束；
- **收紧**：缩小变量可行区间；
- **净化**：改善数值尺度与 LP 几何结构。

# Presolve：规模收缩的定量指标

设原始规模为 $(m_0, n_0, \mathrm{nnz}_0)$，预处理后为 $(m_1, n_1, \mathrm{nnz}_1)$，定义：

$$r_{\mathrm{row}} = 1 - \frac{m_1}{m_0}, \quad r_{\mathrm{col}} = 1 - \frac{n_1}{n_0}, \quad r_{\mathrm{nnz}} = 1 - \frac{\mathrm{nnz}_1}{\mathrm{nnz}_0}.$$

- 行/列/非零缩减率反映 **Presolve 对规模的削减能力**
- 高缩减率（$> 80\%$）：
  - 原模型存在大量冗余约束/变量或非常松的边界
  - Presolve 极大减轻后续 LP/B&B 负担
- 低缩减率（$< 20\%$）：
  - 模型本身已较紧，或 Presolve 对该结构收益有限
  - 若 Presolve time 很长，则可考虑减弱 Presolve

# Presolve：极端情况

```
Presolve: All rows and columns removed
Presolve time: 0.00s
Explored 0 nodes (0 simplex iterations) in 0.00 seconds
Optimal objective 1.234567e+02
```

- 所有约束和变量被预处理消去
- 问题被简化为一个显式的常数解（trivial optimal）
- 对应极易问题：Root/B&B 几乎没有工作量

# Presolve 操作（一）：变量聚合与替代

**1. Aggregating（变量/约束聚合）**
将多条线性约束按代数关系合并，得到更紧/更少的约束

$$x_1 - x_2 = 0 \Rightarrow x_1 \equiv x_2$$

数学本质：线性相关自由度消除；变量并非独立决策维度。

**2. Substituted（变量替代）**
利用等式约束将某些变量用其他变量表示，直接消去

$$x_1 = 5 - x_2$$

数学本质：等式约束给出精确的代数映射，是变量消元（elimination）

**对求解的影响**

- 变量数 $n\downarrow$；非零元 nnz 通常下降；LP / MIP 线性代数规模直接减小。

**分析价值**：

- 反映模型中等式/耦合结构的冗余程度
- 可用于指导上游手工消元、减少维度

# Presolve 操作（二）：固定变量与冗余约束

**1. Fixed Binaries（固定二进制变量）**：通过约束与界推理，二进制变量被固定为 0 或 1
通过约束与变量界推断：

$$x \in \{0,1\}, \quad x \le 0 \;\Rightarrow\; x = 0$$

或通过联合约束：

$$x + y \le 0,\ x, y \ge 0,\ x, y \in \{0,1\} \Rightarrow x = y = 0$$

**2. Redundant Rows（冗余约束）**：被其他约束隐含且不再影响可行域的约束
若约束：

$$a^\top x \le b$$

在变量界下恒成立：

$$\max_{\ell \le x \le u} a^\top x \le b$$

则该约束可被安全删除。

**对求解的影响**

- 整数变量数减少，搜索维度显著下降；
- 约束数 $m \downarrow$，LP 基矩阵规模减小；
- Root LP 与节点 LP 成本同步降低。

# Presolve 操作（三）：变量界收紧

**Tightened Bounds（收紧变量上下界）**：利用线性约束与已知变量界，推导更紧的区间。

$$x + y \leq 10, \quad y \geq 7 \;\Rightarrow\; x \leq 3$$

- 数学本质：区间传播（Interval Propagation）；在线性不等式约束下求变量极值。

**对求解的影响**

- 缩小变量的上下界，可行域显著缩小；
- 分支深度降低；
- Big-M 有效尺度下降，数值条件改善；
- 强化 LP 松弛，Root LP 退化程度减轻。

# Presolve 操作（四）：Probing（逻辑试探）

**基本思想：** 对二进制变量 $x$ 进行假设性赋值

$$x = 0 \quad \text{或} \quad x = 1$$

并在不进入 B&B 的情况下进行快速推理。若产生矛盾或强推导，则可固定变量或收紧界.

**数学本质**

- 有限深度的条件可行性推断；
- 类似 SAT 中的 unit propagation。

**对求解的影响**

- 提前固定 binary 变量；
- 收紧变量界；
- 减少后续分支与回溯。

# Presolve 操作（五）：Implied Bounds（隐含界）

**来源:** 由 binary 与连续变量的联动约束推出的条件界。

**典型:** Big-M 结构

$$y \le Mx, \quad x \in \{0,1\} \Rightarrow \begin{cases} x = 0 \Rightarrow y = 0 \\ x = 1 \Rightarrow y \le M \end{cases}$$

**数学本质**

- 分段可行域分析；
- 条件线性约束下的变量界推导。

**对求解的影响**

- 显著缩小连续变量的“虚假可行区间”；
- 降低 Big-M 诱发的数值病态；
- 改善 Root LP 的几何结构与稳定性。

# Presolve 因果链

**1. Presolve → Root LP**

Presolve 决定：

- Root LP 的 m,n,nnz；变量界紧不紧；是否存在明显退化结构

弱 Presolve ⇒ 大而松的 Root LP ⇒ 慢 + 高度退化

**1. Presolve → Cuts**

Presolve 强：

- 变量界紧; 逻辑结构清晰 → Cuts 更 “精准”

Presolve 弱：

- Cuts 在松弛的大空间中切割; 多而弱，甚至引发数值问题

Presolve 的质量，决定了整个 MIP 求解是否 “在正确的空间中搜索”

# 通过 Presolve 结果反向诊断模型结构（一）

`Presolve removed 0 columns`

**含义:**

- 模型中几乎不存在显式等式关系，或等式写法阻止了代数消元（例如 Big-M 代替等式）
- 人为引入了“虚假的自由度”，导致变量数 n 虚高，nnz 增大，Root LP 维度被放大；

**策略:**

- 尽量用等式替代逻辑联结：
- 能消元的“定义变量”直接消元，不保留中间变量

`Presolve removed 0 rows`

**问题:**

- 约束存在高度重叠但未被形式化识别，或多条约束本质表达同一业务规则
- 约束尺度不同，导致 dominance 判断失败

**策略:**

- 人工检查“族约束”是否可合并；减少“保险型约束”
- 对参数进行缩放，帮助 presolve 识别冗余

# 通过 Presolve 结果反向诊断模型结构（二）

```
Presolve tightened 0 bounds
```

**问题:** 约束未能显著缩小变量的可行区间，Big-M 实际上在整个区间内生效
**策略:**

- 人工推导变量的真实物理/业务界
- Big-M 必须是“问题规模级”的，而不是“安全级”
- 对时间、流量、成本变量显式给出 tight bounds

```
Presolve probing fixed 0 variables
Presolve implied bounds applied 0 times
```

**问题:** binary 与连续变量之间的联动太弱，或 Big-M 写法阻断了逻辑传播
**策略:**

- 强化 binary $\rightarrow$ continuous 的因果关系
- 用 indicator constraints 代替 Big-M
- 避免“x 控制 y 的系数”，而是“x 控制 y 的存在性”

# Presolve 检测不可行（Infeasible）

```
Presolve: model is infeasible
Presolve time: 0.05s
```

**特征**：

- B&B 尚未开始，直接判定无可行解
- 后续日志通常是简单的 infeasible summary

**价值**：

- 数据/建模质量监控：一旦比例异常升高，提示 pipeline 问题
- 可结合 IIS（不可约不一致子系统）进行诊断

# Presolve 检测无界（Unbounded）与 Trivial 解

**无界示例**：

```
Presolve: model is unbounded
Presolve time: 0.01s
```

**Trivial 解示例**：

```
Presolve: All rows and columns removed
Presolve time: 0.00s
Optimal objective 1.234567e+02
```

**分析价值**：

- Infeasible / Unbounded：指向建模或数据错误
- Trivial：表明实例极简单，可从数据生成层直接过滤

# Presolve 时间与占比

**典型字段**：

```
Presolve time: 6.57s
...
Explored 84723 nodes (78138699 simplex iterations) in 7200.00 seconds
```

设总求解时间为 $T_{\text{total}}$，预处理时间为 $T_{\text{pre}}$，定义：

$$\rho_{\text{pre}} = \frac{T_{\text{pre}}}{T_{\text{total}}}.$$

- $\rho_{\text{pre}}$ 小且规模缩减大：
  - Presolve 极具性价比，应保留甚至增强
- $\rho_{\text{pre}}$ 大且缩减率低：
  - 对该类实例可考虑减弱 Presolve 或调整相关参数

# 数值相关提示（与 Presolve 相关）

- 常见数值提示（可能出现在 Presolve / Root 交界处）：
  - 数值不稳定（numerical trouble）
  - 多次 scaling / re-scaling
- 与 Header 中的 `Coefficient statistics` 联动：
  - 矩阵 / RHS 范围跨越多个数量级 $\rightarrow$ Presolve 可能需要更激进的缩放
  - 极端 big-M $\rightarrow$ 容易在预处理阶段触发数值修正
- 结合 Root LP 行为验证 presolve 质量
  - presolve 弱 + root LP simplex iterations 巨大

$$\Rightarrow \text{建模问题，而不是参数问题}$$

- 二次开发方向：
  - 将“数值风险”打标，作为后续建模优化的输入
  - 针对高风险实例启用定制的 scaling / Presolve 策略

# 可提炼的 Presolve 指标

从日志中可直接构造的指标示例：

- 规模相关：
  - 行/列/非零缩减率 $r_{\text{row}}, r_{\text{col}}, r_{\text{nnz}}$
- 时间相关：
  - 预处理时间 $T_{\text{pre}}$
  - 占比 $\rho_{\text{pre}} = T_{\text{pre}} / T_{\text{total}}$
- 状态标签：
  - `status` $\in \{\text{normal, infeasible\_in\_presolve, unbounded\_in\_presolve, trivial}\}$
- 数值风险：
  - 是否出现 numerical trouble / 多次 scaling

# Presolve 信息的工程价值

- **建模层面**：
  - 对高缩减率实例，分析被删/被固定的变量与约束，优化建模结构
  - 针对 numeric risky 实例，调整 big-M、单位尺度、界设置
- **求解策略层面**：
  - 依据缩减效果与时间开销调整 Presolve 相关参数
  - 针对易 infeasible 的实例，自动触发 IIS 提取与诊断流程
- **自动化/监控层面**：
  - 在批量求解平台中记录 Presolve 指标，做实例族画像
  - 后续可基于这些特征训练“Presolve 策略推荐器”

# 本章小结：Presolve 的视角

- Presolve 是从“原始模型”到“可解模型”的关键桥梁
- 通过日志可精确观察：
  - 模型规模如何缩减
  - 哪些变量/约束被固定、删除或收紧
  - 是否在预处理阶段就发现 infeasible / unbounded / trivial
  - 预处理本身耗费了多少时间、引发了哪些数值修正
- Presolve 日志可直接转化为：
  - 建模改进建议
  - 求解策略调参依据
  - 自动化监控与智能调度的输入特征