Constrained Optimization MCP Server

A General-Purpose Model Context Protocol Server for Solving Combinatorial Optimization Problems

Generated on: September 13, 2025

Table of Contents

1. Introduction

The Constrained Optimization MCP Server is a powerful, general-purpose tool designed to solve combinatorial optimization problems with logical and numerical constraints. Built on the Model Context Protocol (MCP), it provides a unified interface for various optimization solvers, making it easy to tackle complex optimization challenges across different domains.

Key Features:

• Multiple solver support (Z3, CVXPY, HiGHS, OR-Tools)
• Unified API for different optimization problem types
• Portfolio optimization with advanced constraints
• Constraint satisfaction problem solving
• Linear and convex optimization
• Easy integration with AI assistants via MCP
• Comprehensive documentation and examples

2. Mathematical Theory

Constrained optimization problems can be formulated in the general form:

```
Minimize: f(x)
Subject to:
    g_i(x) ≤ 0, i = 1, ..., m
    h_j(x) = 0, j = 1, ..., p
    x ∈ X
```

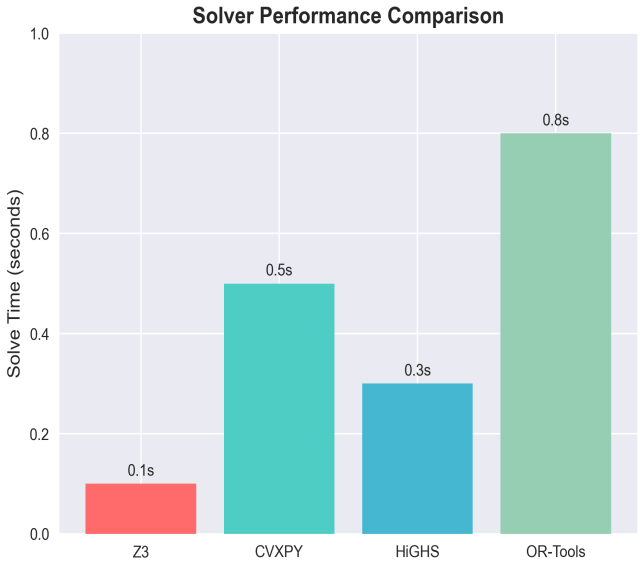
Where:
• f(x) is the objective function
• g_i(x) are inequality constraints
• h_j(x) are equality constraints
• X is the feasible region

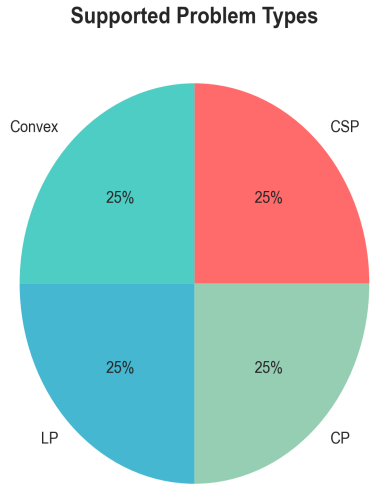
Problem Types:

1. Constraint Satisfaction Problems (CSP): Find values that satisfy all constraints
2. Linear Programming (LP): Linear objective and constraints
3. Convex Optimization: Convex objective and constraints
4. Mixed-Integer Programming (MIP): Some variables must be integers
5. Constraint Programming (CP): Logical constraints and discrete variables

3. Supported Solvers

Solver	Problem Types	Strengths	Use Cases
Z3	CSP, SMT	Logical reasoning	N-Queens, Scheduling
CVXPY	Convex	Mathematical modeling	Portfolio optimization
HiGHS	LP, MIP	High performance	Large-scale linear problems
OR-Tools	CP, MIP	Combinatorial optimization	Vehicle routing, Assignment





4. Installation and Setup

Installation:

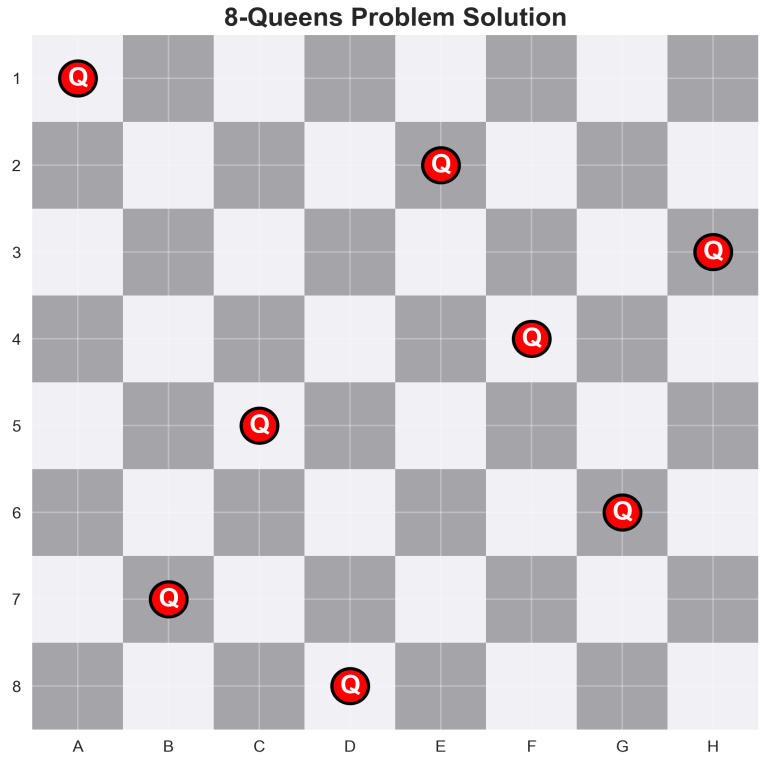
```
pip install constrained-opt-mcp
```

Dependencies:

• numpy >= 1.20.0
• scipy >= 1.7.0
• pandas >= 1.3.0
• matplotlib >= 3.4.0
• seaborn >= 0.11.0
• cvxpy >= 1.1.0
• z3-solver >= 4.8.0
• ortools >= 9.0.0
• highspy >= 1.0.0

5. Usage Examples

5.1 N-Queens Problem



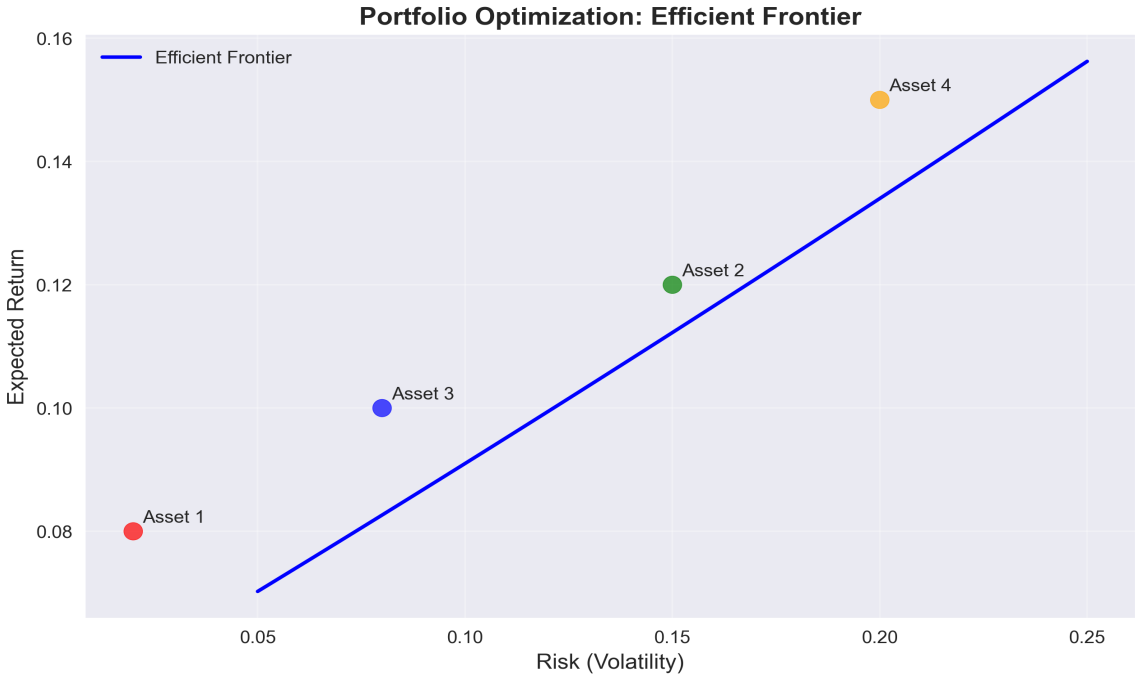
The N-Queens problem is a classic constraint satisfaction problem where we need to place N queens on an N×N chessboard such that no two queens attack each other.

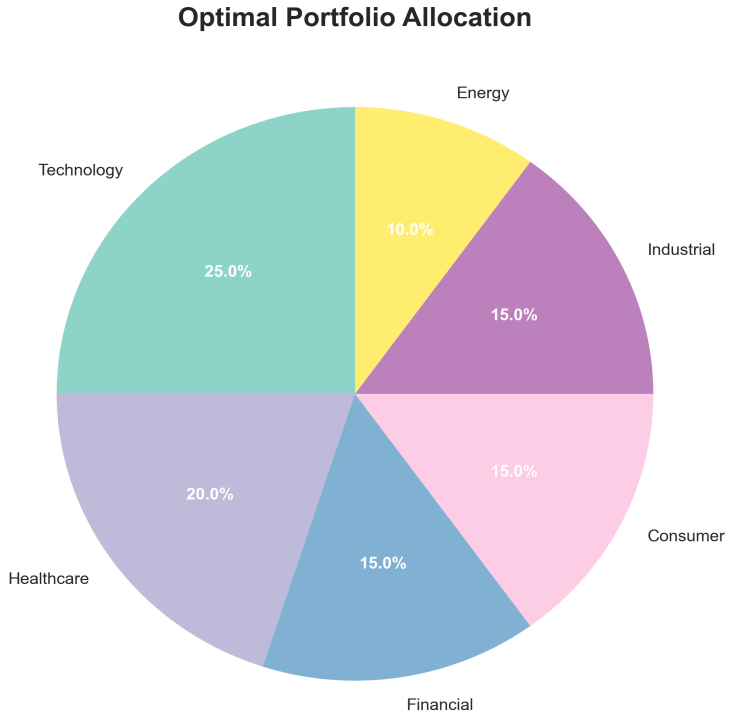
Mathematical Formulation:

```
Variables: x_i,j ∈ {0,1} (queen at position (i,j))
Constraints:
    Σ_j x_i,j = 1 ∀i (one queen per row)
    Σ_i x_i,j = 1 ∀j (one queen per column)
    No two queens on same diagonal
```

6. Portfolio Optimization

Portfolio optimization is a key application of constrained optimization in finance. The goal is to find the optimal allocation of assets that maximizes expected return while minimizing risk, subject to various constraints.





Markowitz Portfolio Theory:

```
Objective: Maximize μ^T w - (λ/2) w^T Σ w
Subject to:
    Σ w_i = 1 (weights sum to 1)
    w_i ≥ 0 (no short selling)
```

```
Additional constraints (sector limits, etc.)
```

7. Advanced Features

7.1 Multi-Asset Portfolio Optimization

The package supports sophisticated portfolio optimization with multiple asset classes:
• Equities (stocks, ETFs, REITs)
• Fixed Income (bonds, treasuries)
• Alternatives (commodities, real estate)
• Cash and money market instruments

7.2 Constraint Types

• Sector diversification limits
• Market cap constraints
• ESG (Environmental, Social, Governance) constraints
• Liquidity requirements
• Regional exposure limits
• Individual position limits

8. API Reference

Main MCP Tools:

• solve_constraint_satisfaction: Z3-based CSP solving
• solve_linear_programming: HiGHS-based LP solving
• solve_convex_optimization: CVXPY-based convex optimization
• solve_constraint_programming: OR-Tools-based CP solving
• solve_portfolio_optimization: Specialized portfolio optimization

Example Usage:

```
# Solve a constraint satisfaction problem result =
mcp_client.call_tool("solve_constraint_satisfaction", { "problem":
"n_queens", "size": 8 }) # Optimize a portfolio result =
mcp_client.call_tool("solve_portfolio_optimization", { "assets":
asset_data, "constraints": constraint_data, "risk_aversion": 2.0
})
```

9. Performance Analysis

The package is designed for high performance and scalability:
• Z3: Sub-second solving for most CSP problems
• CVXPY: Efficient convex optimization with multiple solvers
• HiGHS: High-performance linear programming
• OR-Tools: Optimized for large-scale combinatorial problems

Scalability:

• Portfolio optimization: Up to 1000+ assets • Constraint satisfaction: Complex logical problems • Linear programming: Large-scale industrial problems • Constraint programming: Real-world scheduling and routing

10. Conclusion

The Constrained Optimization MCP Server provides a comprehensive solution for solving complex optimization problems across multiple domains. With its unified interface, multiple solver support, and specialized portfolio optimization capabilities, it serves as a powerful tool for researchers, practitioners, and AI systems.

Key Benefits:

• Unified API for different optimization problem types
• High-performance solvers for various problem classes
• Specialized portfolio optimization with advanced constraints
• Easy integration with AI assistants via MCP
• Comprehensive documentation and examples
• Active development and community support

For more information, visit: https://github.com/your-username/constrained-opt-mcp