Branch-and-Price Multiple Knapsack Toolkit

December 15, 2025

1 Overview

This codebase implements a branch-and-price solver for the Multiple Knapsack Problem (MKP) inspired by Lalonde et al. (2022). It provides classic and L2 formulations, a Dantzig–Wolfe master with column generation, and a branch-and-price loop with simple no-good cuts and a CP-SAT repair for fractional assignments. Benchmarks and plots are included for the SMALL and Falkenauer (FK) instance families.

2 Formulations and Algorithm

- **L2 relaxation (eqs. 13–18):** binary t_j for item selection, binary x_{ij} for assignment, with knapsack capacities, linking constraints, and aggregated capacity $\sum_j w_j t_j \leq \sum_i c_i$.
- **Dantzig–Wolfe master (eqs. 28–33):** pattern pools P_0 (aggregated capacity) and P_i (per bin) with variables y_a plus dual-cut variables s_j. Constraint (31) is present but the upper bound defaults to total profit and is not tightened during the search.
- **Pattern seeding:** empty and single-item patterns, multiple greedy patterns by ratio/profit/weight, and a core DP-derived pattern per pool.
- **Column generation:** restricted master built via `DWMasterLPBuilder`, solved with OR-Tools LP (GLOP). Duals feed knapsack pricing for P_0 and each P_i; profitable columns are added unless duplicate or incompatible with branching filters. No stabilization, dual-bound stopping, or subgradient phase is implemented.
- **Branch-and-price:** best-first on node upper bounds. Branch only on fractional t_j. If t is integral but x is fractional, a CP-SAT feasibility check (VSBPP-SAT) attempts to repair assignments; infeasible selections add a no-good cut. Cuts are simple set forbiddances (no $E(S)$ strengthening).

3 Comparison to Lalonde et al. (2022)

- No instance reduction or MULKNAP warm-start; the paper uses both as preprocessing and incumbent tightening.
- Constraint (31) is static (total profit) and never updated to the incumbent floor, weakening the integrality push noted in the paper.

- Column generation omits dual bound checks (eq. 44), subgradient-based column generation when stalled, and any stabilization. Pricing ignores duals from no-good cuts and does not use the $E(S)$ strengthening (eq. 35).

- Branching rule is "most fractional t_j" only; the paper limits candidates and picks the largest-impact variable, and applies dominance filtering and Lagrangian probing to fix variables aggressively.

- Packing subproblem differs: the paper uses a staged heuristic/set-packing/arc-flow pipeline that can prove infeasibility; the implementation uses a single CP-SAT feasibility attempt and treats UNKNOWN by adding a cut.

- Tooling differs (OR-Tools LP/CP-SAT vs. CPLEX), so numerical behavior and performance bounds are not directly comparable.

4 Experiments

Configuration unless noted: time limit 600 s/instance, max nodes 1000, OR-Tools LP, default pattern seeding, branch-and-price with gap tolerance as stated.

Dataset/config	Total	Optimal	Gap-limit	Avg gap	Avg time (s)	Notes
SMALL, gap 1%	179	105	74	0.000974	12.745	From `time-limit600_gap0.`
SMALL, gap 0%	179	179	0	0.000000	16.468	Tighter gap; slower
SMALL, gap 0%, 10 inst.	10	10	0	0.000000	0.092	Demo subset run
FK_1, first 10, gap 0%	10	10	0	0.000000	8.487	From `...max-instances10`

All detailed CSV/JSON outputs are under `benchmark_results/`, with plots and summaries in `analysis_out/`.

5 Future Improvements

- Tighten constraint (31) with the best incumbent and carry its dual into pricing.

- Add dual-bound checks, stabilization, and subgradient-based column generation for stalled nodes.

- Implement dominance filtering and Lagrangian probing; adopt the paper's branching candidate selection.

- Strengthen no-good cuts with $E(S)$ and propagate their duals into P_0 pricing.

- Replace the CP-SAT-only VSBPP-SAT with the paper's staged heuristic/set-packing/arc-flow pipeline to prove infeasibility more often.