Výpočtové riešenie úloh o dláždení polyominami

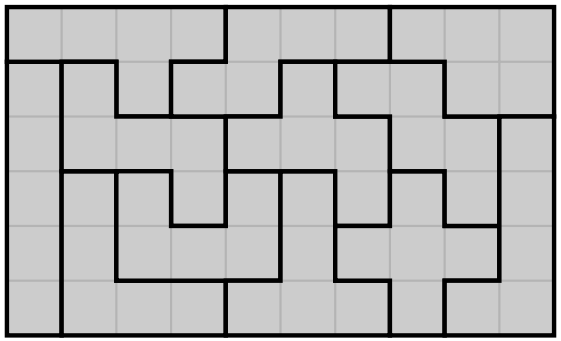
Bakalárska práca

Dávid Mišiak
doc. RNDr. Ján Mazák, PhD.

Katedra Informatiky
Fakulta Matematiky, Fyziky a Informatiky
Univerzita Komenského

Problém dláždenia

Dá sa mriežka 10×6 *pokryť všetkými* 12 *pentominami? Každé musí byť použité práve raz, prevracanie je dovolené.*



- vstup: tvar mriežky, tvary a počty dielikov, prevracanie
- výstup: ÁNO / NIE, (príklad vydláždenia)
- vo všeobecnosti NP-úplný problém

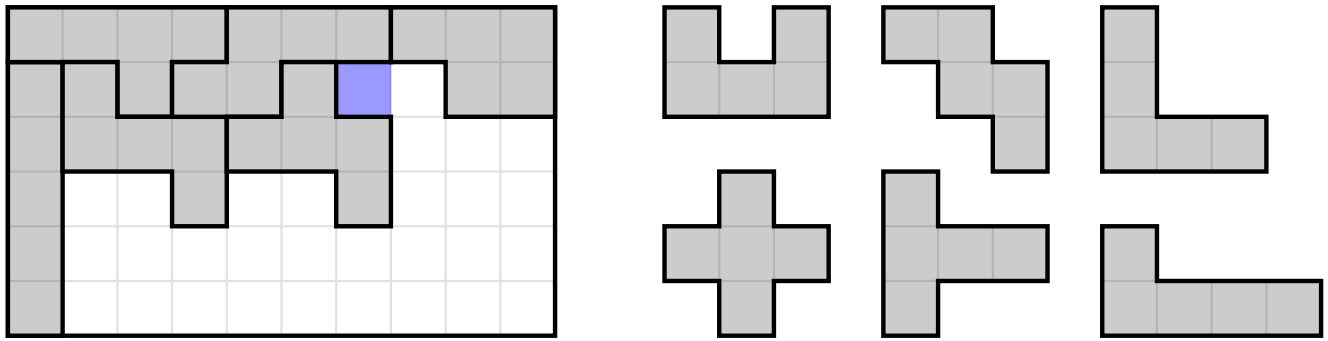
Cieľ

Cieľ práce: Implementovať viacero algoritmov (CLI nástroj), porovnať ich na rôznych vstupoch a identifikovať zaujímavé triedy vstupov.

Algoritmy:

- jednoduchý backtracking
- Algoritmus X
- konverzia na SAT
- konverzia na ILP
- konverzia na CSP

Jednoduchý backtracking



- rekurzívne skúšame obsadiť prázdne políčko vľavo hore všetkými zostávajúcimi dielikmi
- rôzne stratégie výberu dieliku

Algoritmus X

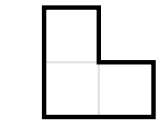
- rieši problém presného pokrytia (exact cover)
  - množina prvkov M a kolekcia podmnožín, ktoré majú presne pokryť M
  - prvky = políčka a dieliky; podmnožiny = umiestnenia dielikov
- D. Knuth, technika *Dancing links*
- často používaný na dláždenie (napr. SageMath), v princípe podobný nášmu backtrackingu
- vyladený, ale nevieme vyjadriť viac inštancií toho istého dieliku

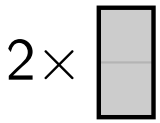
Konverzia na SAT

- premenné = umiestnenia dielikov
- kódovanie EO a AMK obmedezení
  - pre každé políčko: EO(*umiestnenia, ktoré ho pokrývajú*)
  - pre každý dielik: AMK(*jeho umiestnenia*)
- viacero variantov konverzie, optimalizácie
- PBLib na implementácie EO a AMK
- CaDiCaL, CryptoMiniSat

Konverzia na ILP

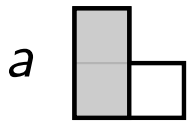
- binárne premenné = umiestnenia dielikov
- konverzia podobná ako pri SAT, ale máme natívne EO a AMK
- COIN-BC, Gurobi



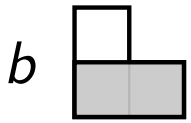


2×

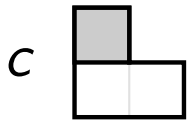
1×



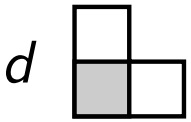
a



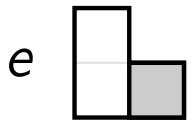
b



c



d



e

$$a + c = 1$$
$$a + b + d = 1$$
$$b + e = 1$$

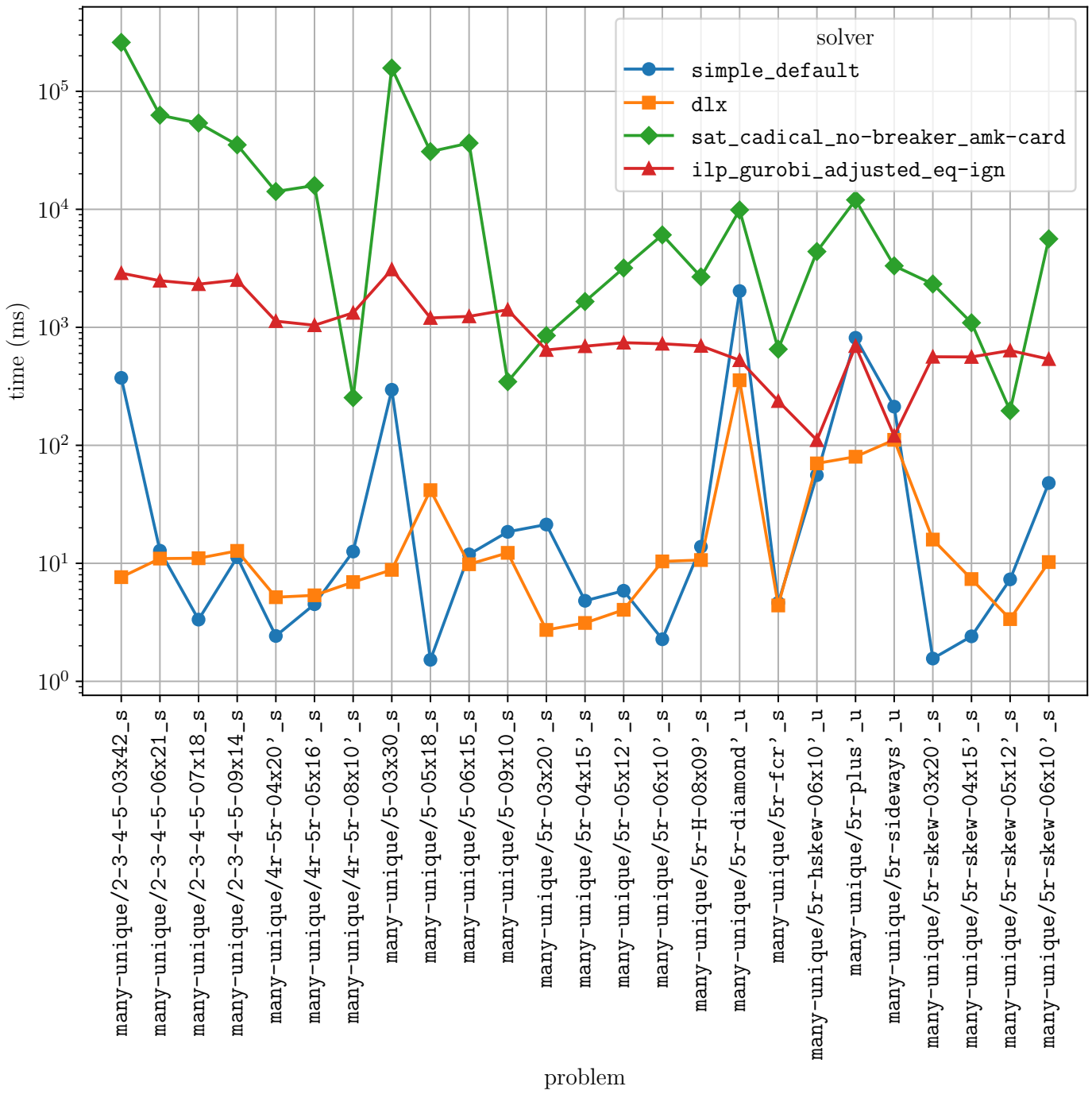
$$a + b \leq 2$$
$$c + d + e \leq 1$$

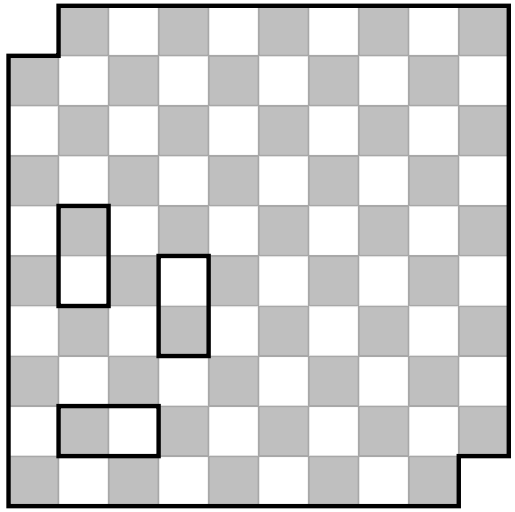
Konverzia na CSP

- constraint satisfaction problems – expresívnejšie obmedzenia než pri SAT alebo ILP
- jazyk a nástroj MiniZinc
  - preklad do jazyka FlatZinc + podkladový CSP solver (Gecode, Chuffed, Gurobi, . . .)
- obmedzenie `geost` znemožní prekryvy
  - definície tvarov, povolené tvary pre objekty (umiestnené dieliky)
  - premenné = tvary a súradnice objektov
- prístup sa ukázal byť nevhodný

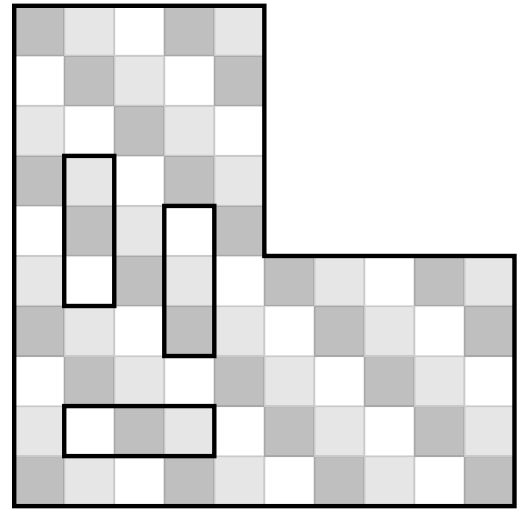
Vstupy a výsledky meraní

- zbierka vstupov – rôzne typy, rôzne veľkosti mriežky
- na základe výsledkov sme vybrali reprezentantov algoritmov a detailne ich porovnali
  - Prípadová štúdia: Veľa rôznych dielikov
  - Prípadová štúdia: Ofarbovacie invarianty

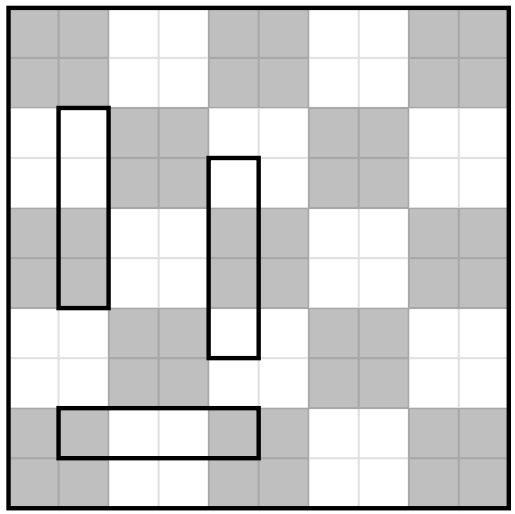

solver
simple_default
dlx
sat_cadical_no-breaker_amk-card
ilp_gurobi_adjusted_eq-ign
time (ms)
10^0
10^1
10^2
10^3
10^4
10^5
many-unique/2-3-4-5-03x42_s
many-unique/2-3-4-5-06x21_s
many-unique/2-3-4-5-07x18_s
many-unique/2-3-4-5-09x14_s
many-unique/4r-5r-04x20'_s
many-unique/4r-5r-05x16'_s
many-unique/4r-5r-08x10'_s
many-unique/5-03x30_s
many-unique/5-05x18_s
many-unique/5-06x15_s
many-unique/5-09x10_s
many-unique/5r-03x20'_s
many-unique/5r-04x15'_s
many-unique/5r-05x12'_s
many-unique/5r-06x10'_s
many-unique/5r-H-08x09'_s
many-unique/5r-diamond'_u
many-unique/5r-fcr'_s
many-unique/5r-hskew-06x10'_u
many-unique/5r-plus'_u
many-unique/5r-sideways'_u
many-unique/5r-skew-03x20'_s
many-unique/5r-skew-04x15'_s
many-unique/5r-skew-05x12'_s
many-unique/5r-skew-06x10'_s
problem



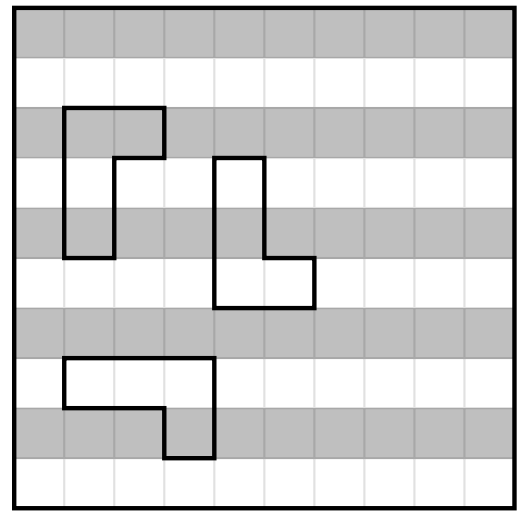
(a)



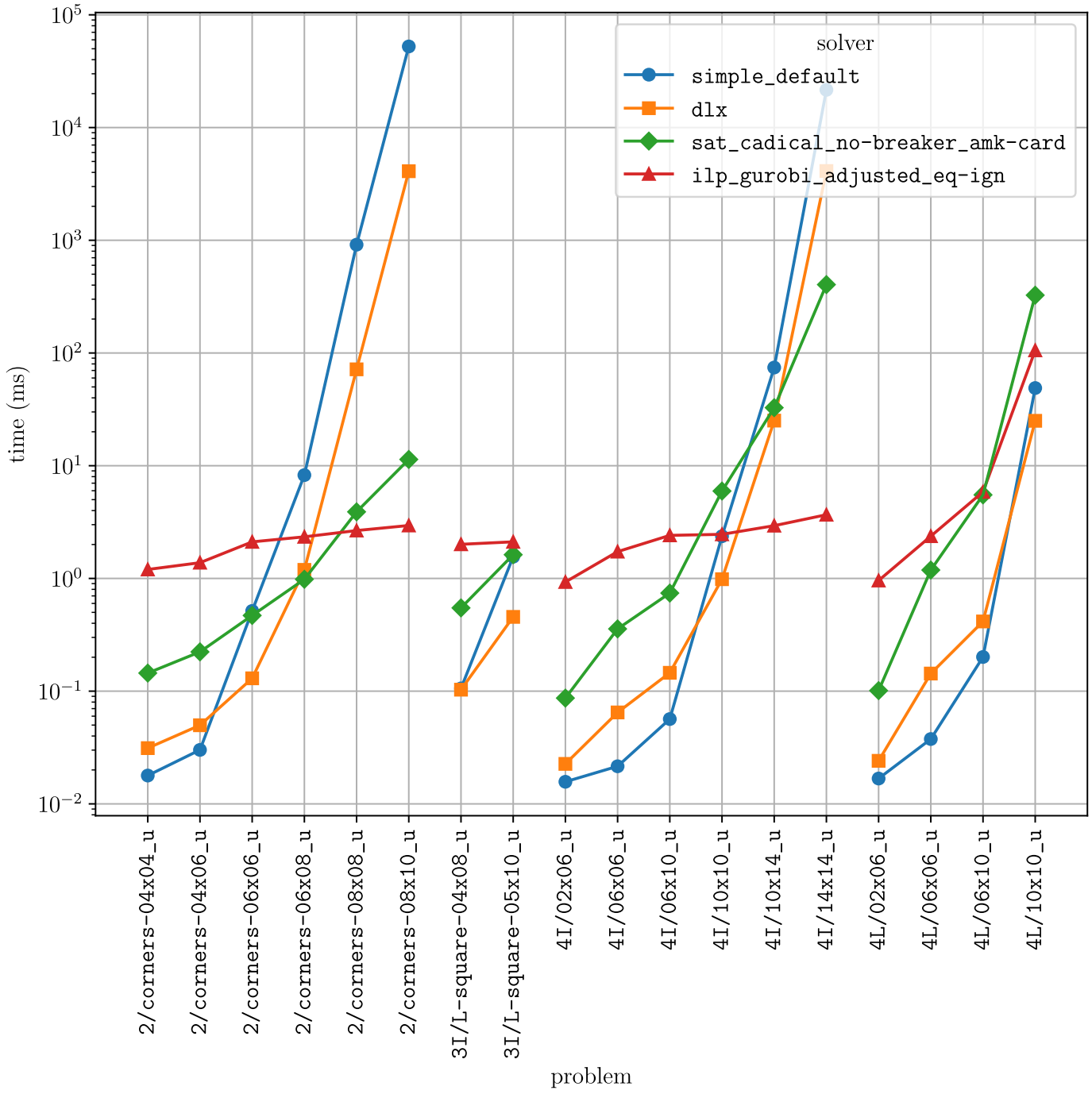
(b)



(c)



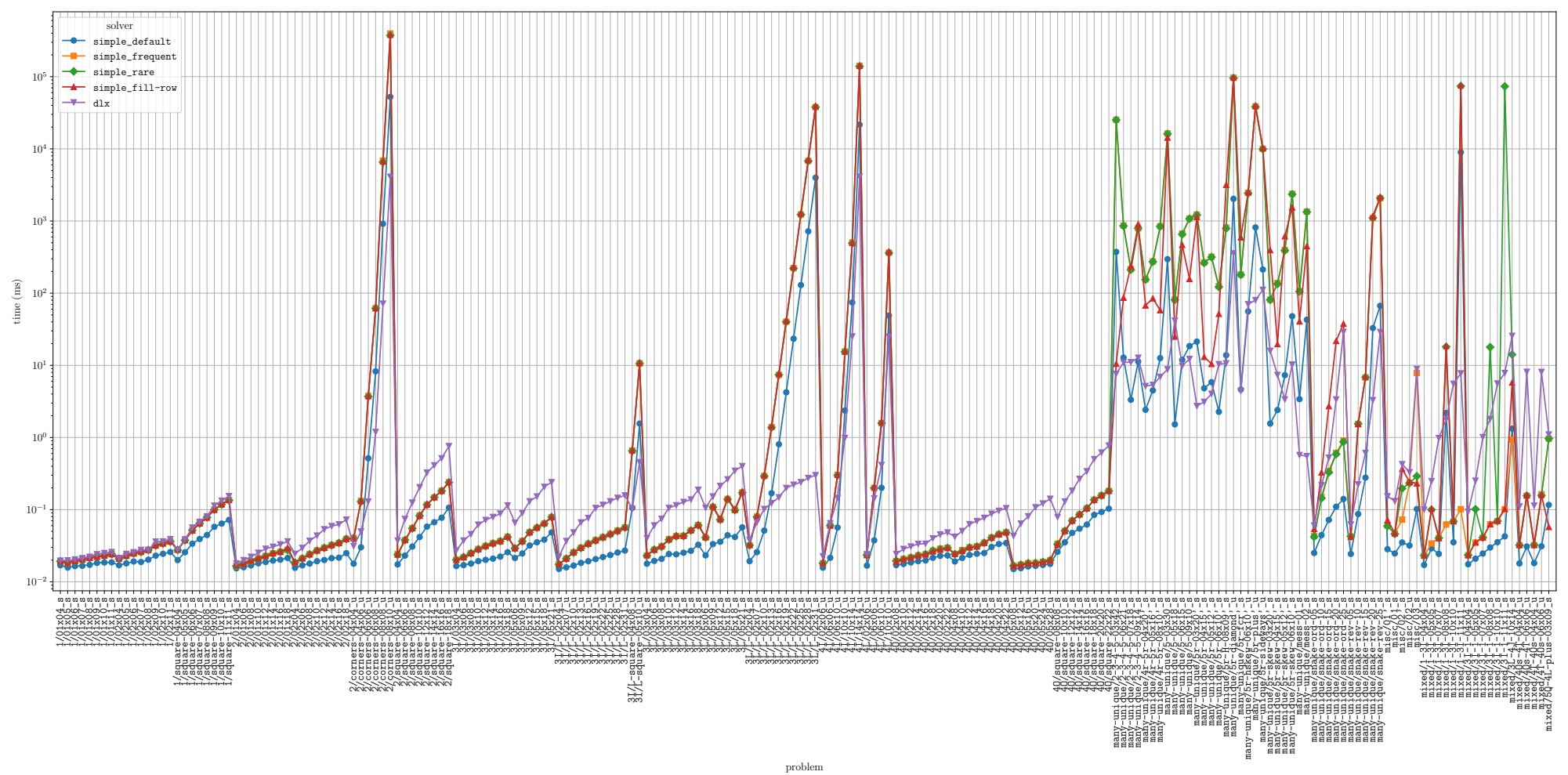
(d)


solver
simple_default
dlx
sat_cadical_no-breaker_amk-card
ilp_gurobi_adjusted_eq-ign
time (ms)
problem
10^{5}
10^{4}
10^{3}
10^{2}
10^{1}
10^{0}
10^{-1}
10^{-2}
2/corners-04x04_u
2/corners-04x06_u
2/corners-06x06_u
2/corners-06x08_u
2/corners-08x08_u
2/corners-08x10_u
3I/L-square-04x08_u
3I/L-square-05x10_u
4I/02x06_u
4I/06x06_u
4I/06x10_u
4I/10x10_u
4I/10x14_u
4I/14x14_u
4L/02x06_u
4L/06x06_u
4L/06x10_u
4L/10x10_u

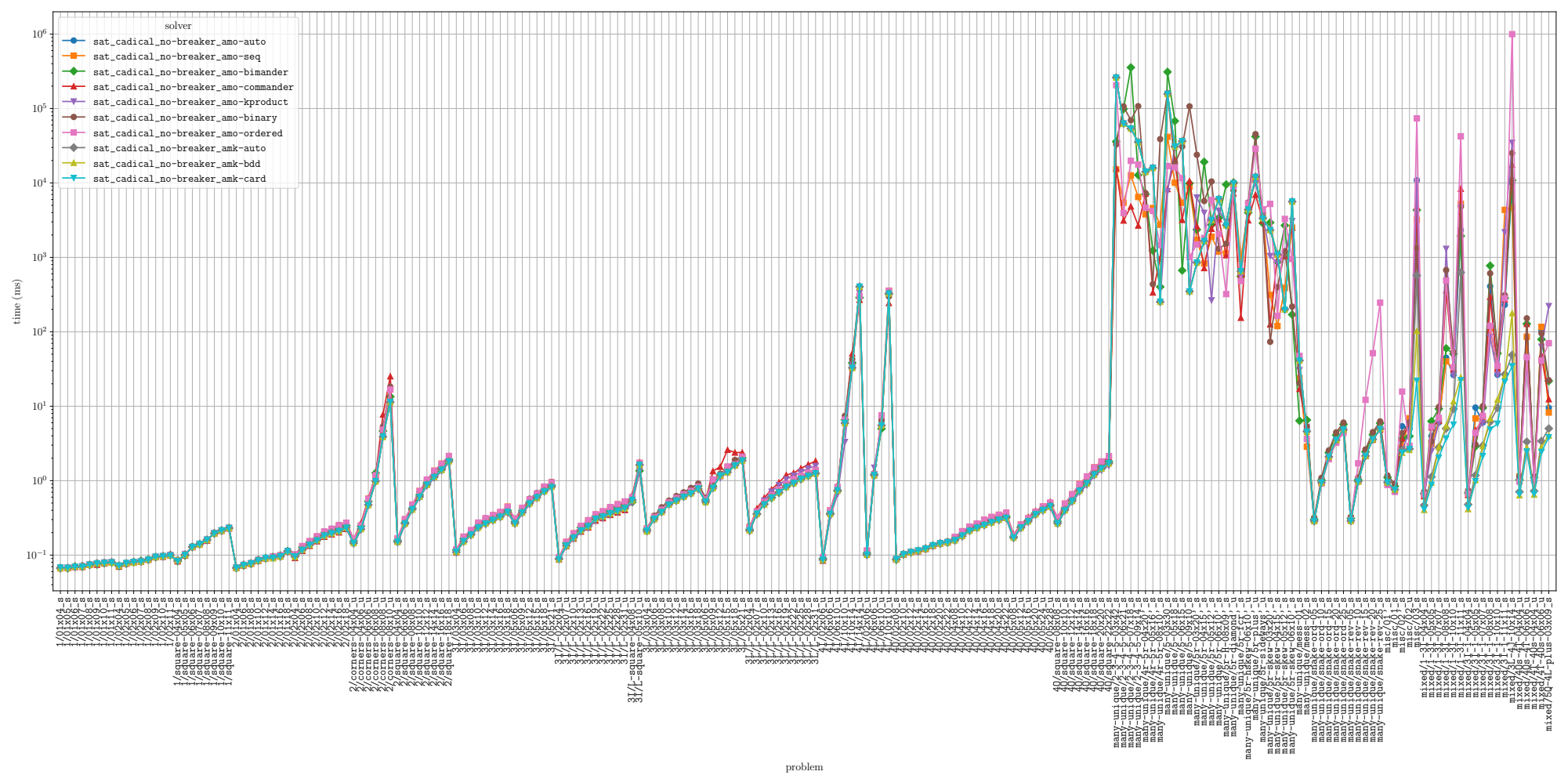
Záver

- rôzne solvery sú vhodnejšie na rôzne typy vstupov
- ILP dokáže odhaliť niektoré ofarbovacie invarianty
- ďalší výskum:
  - hlbšie preskúmať výkony jednotlivých algoritmov a zlepšiť ich
  - vyskúšať iné CSP solvery a konverzie
  - navrhnúť algoritmus, ktorý je schopný využiť aj zložitejšie ofarbovacie invarianty

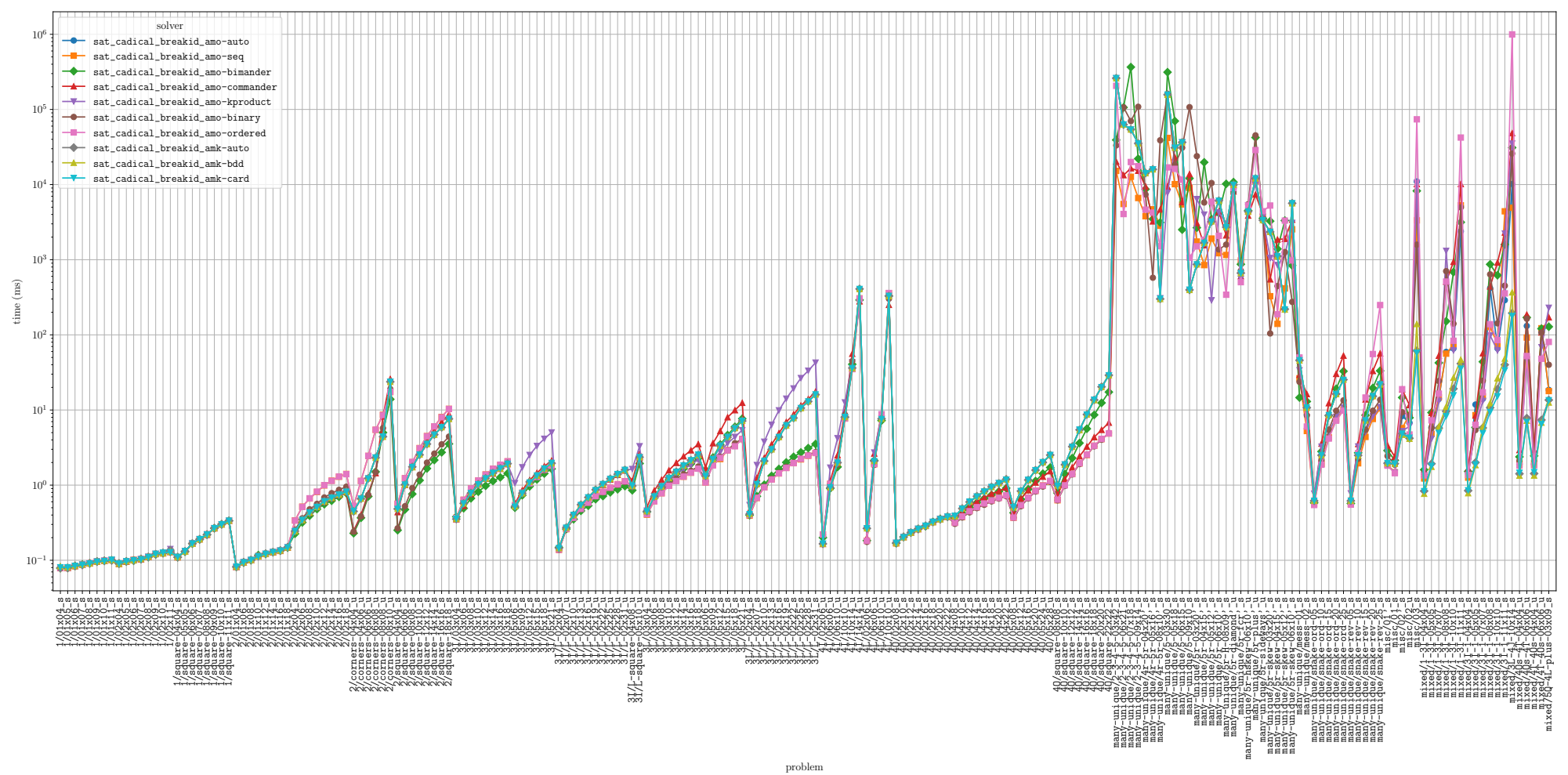
Prílohy



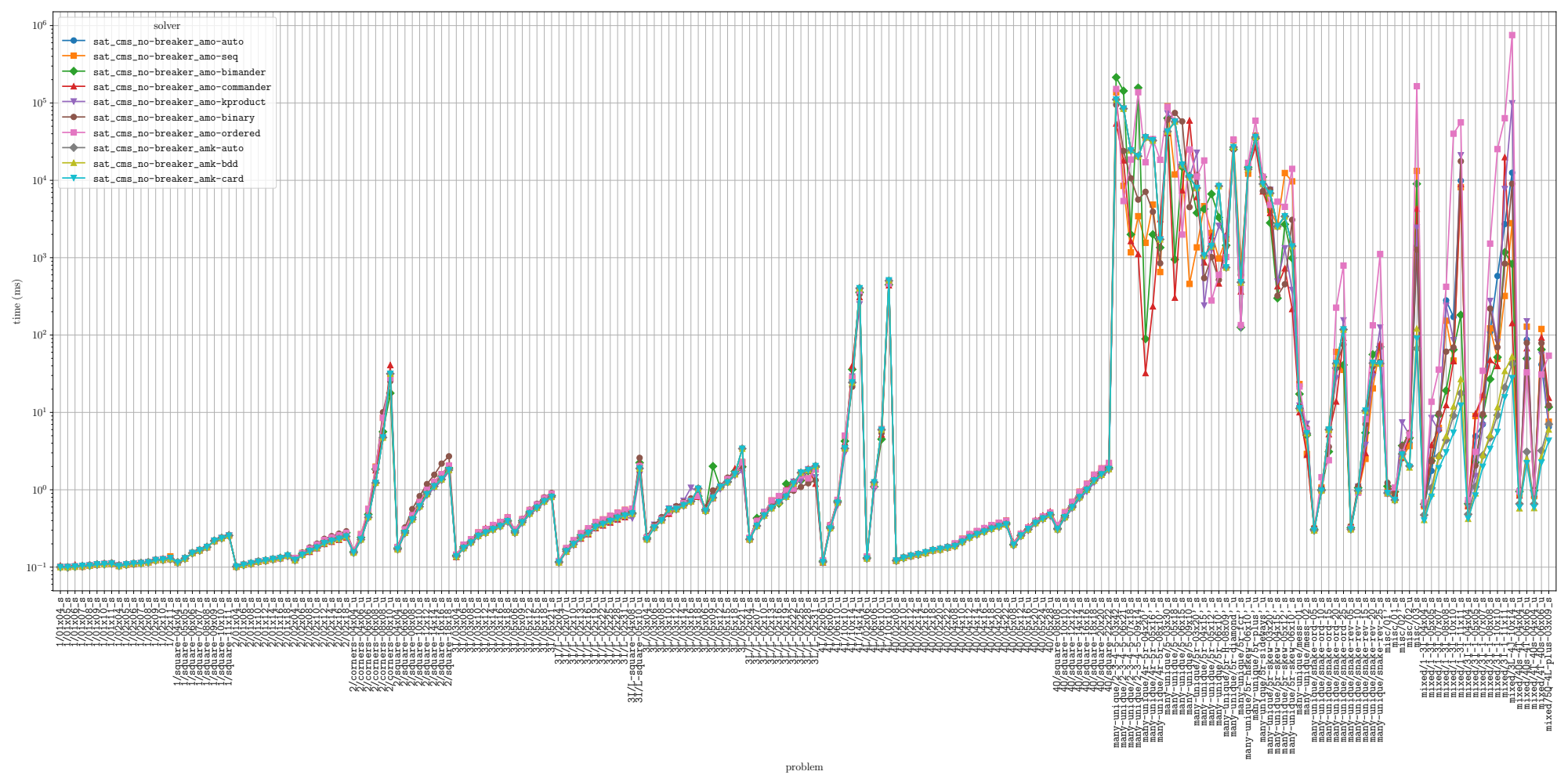
01-simple-dlx



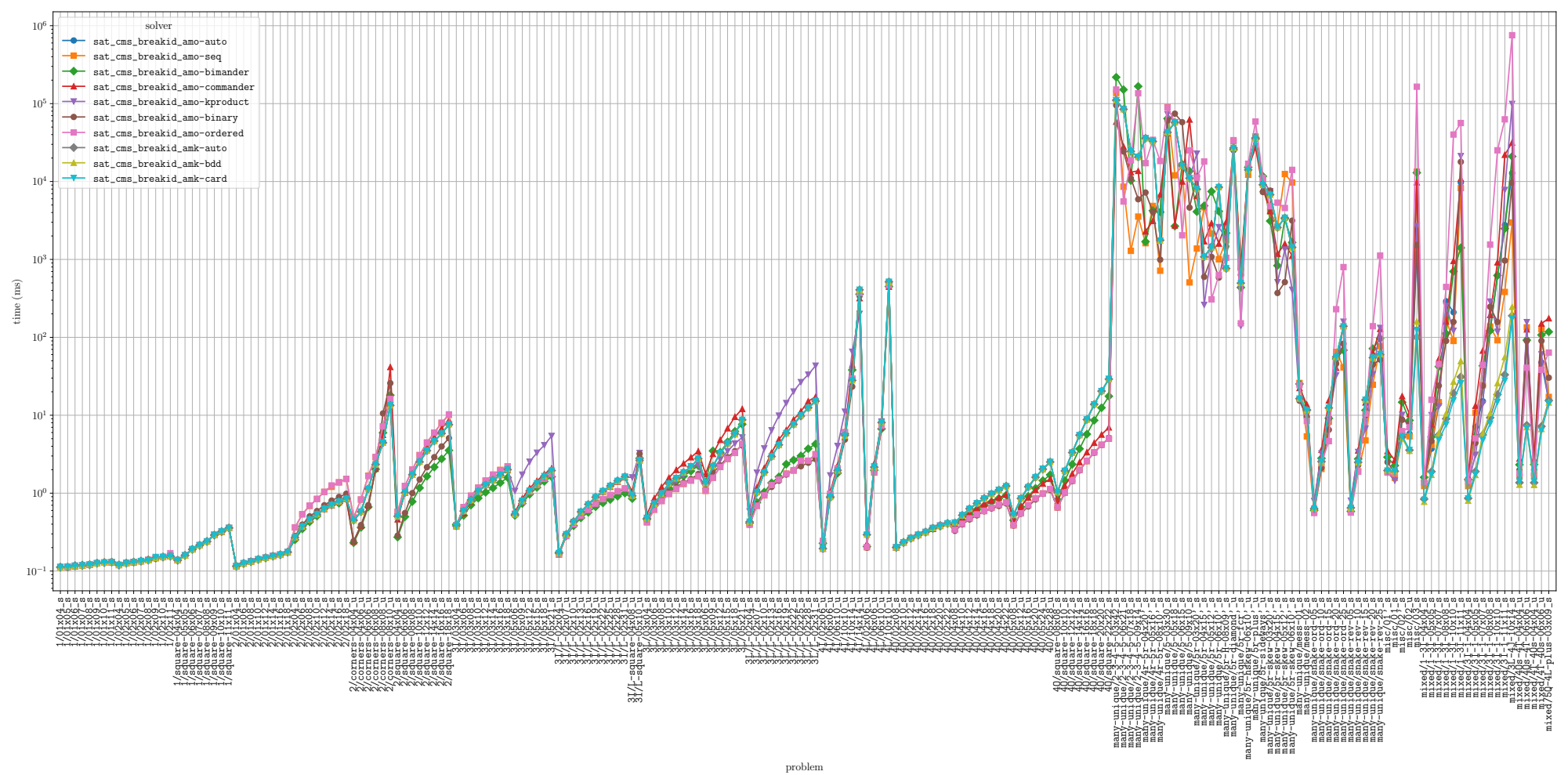
02-sat-cadical-no-breaker



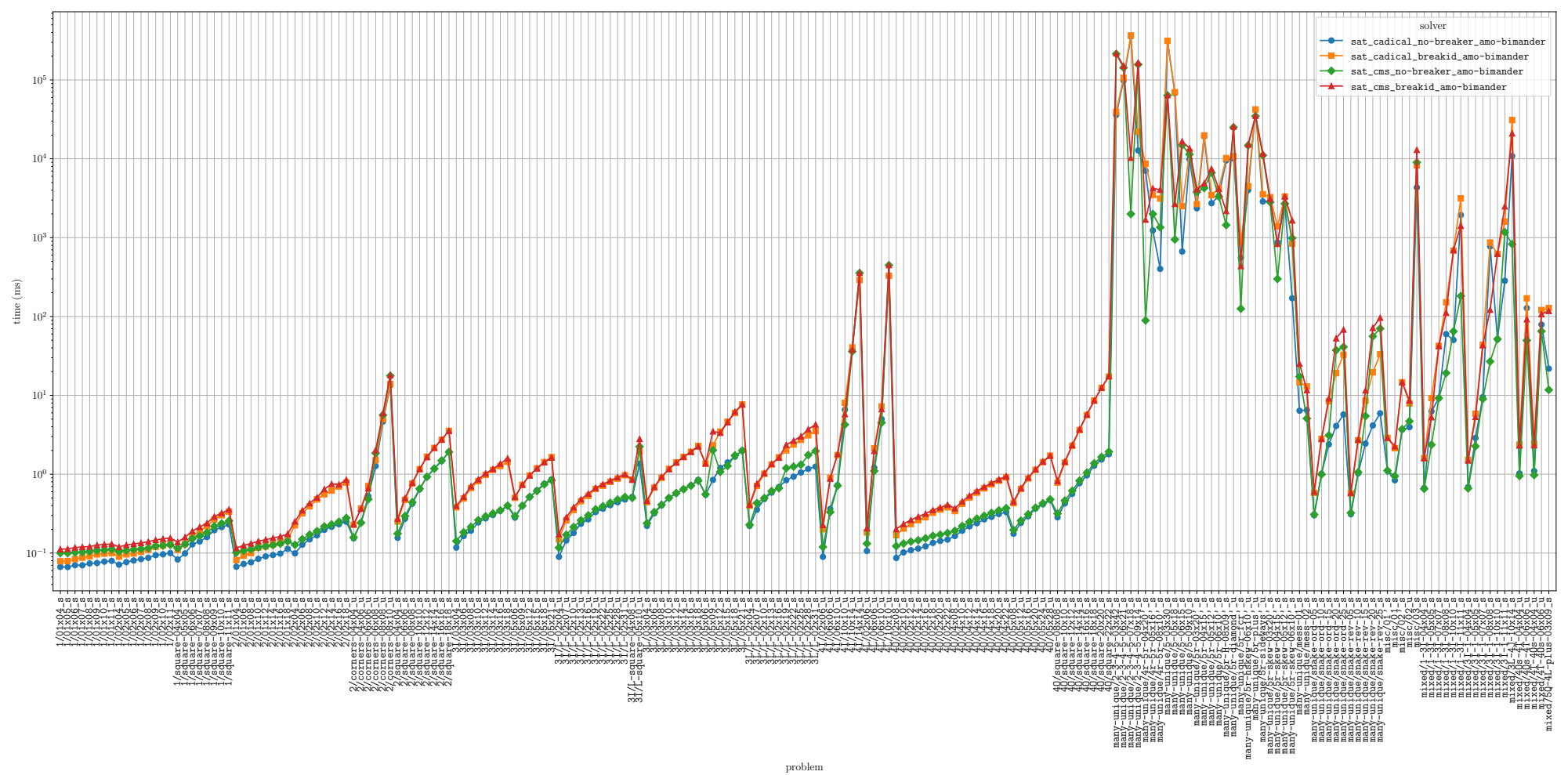
03-sat-cadical-breakid



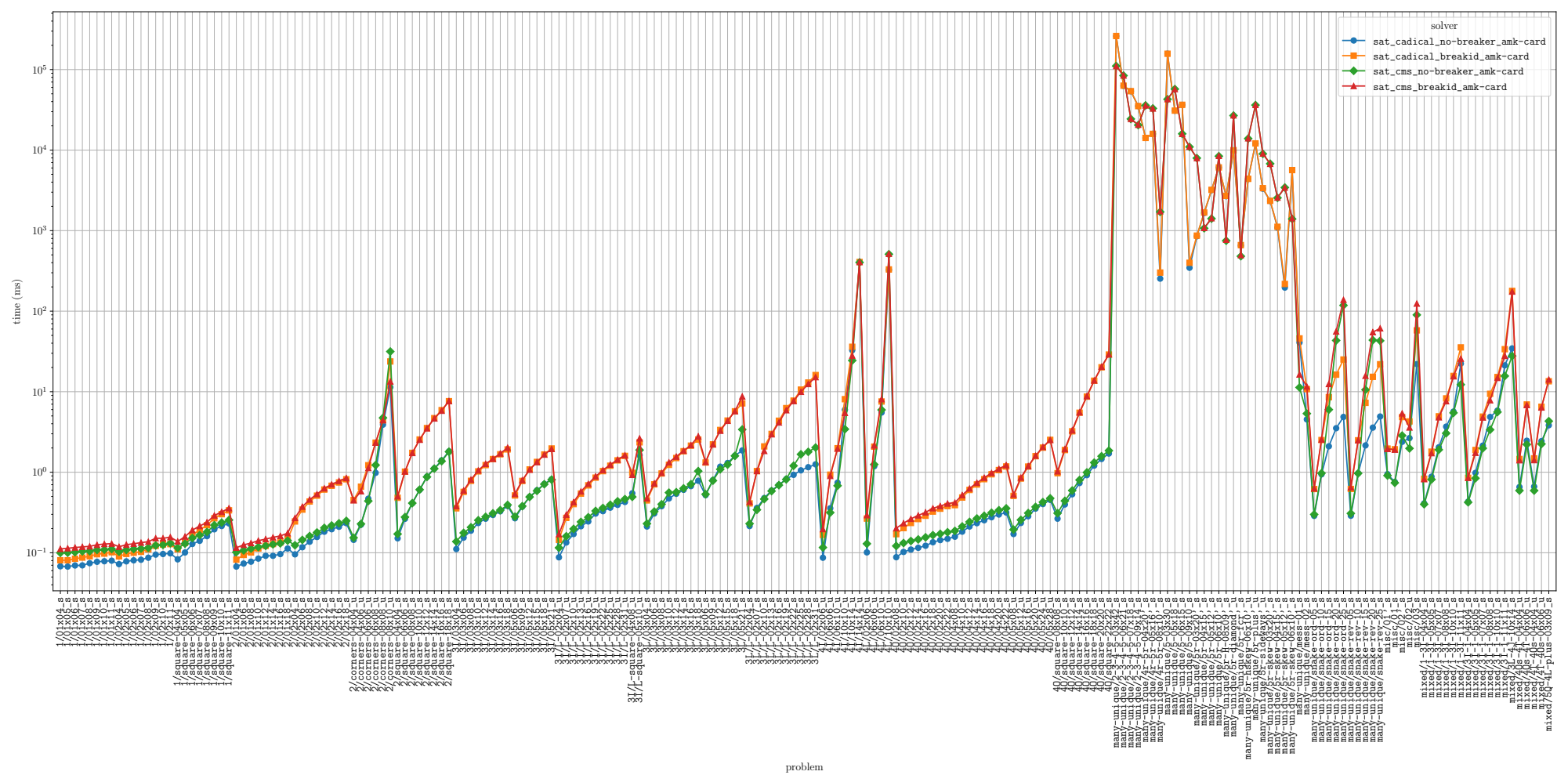
04-sat-cms-no-breaker



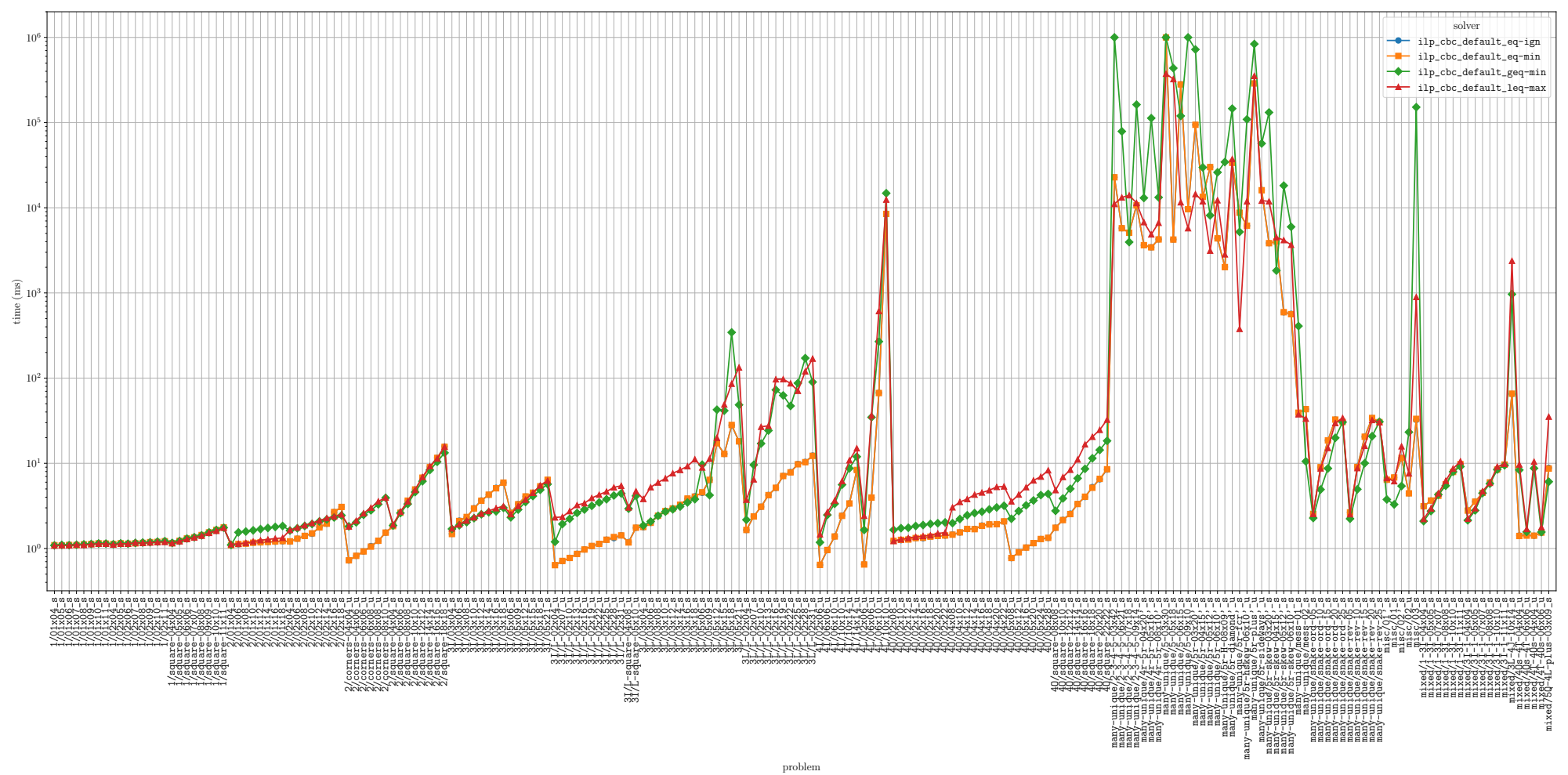
05-sat-cms-breakid



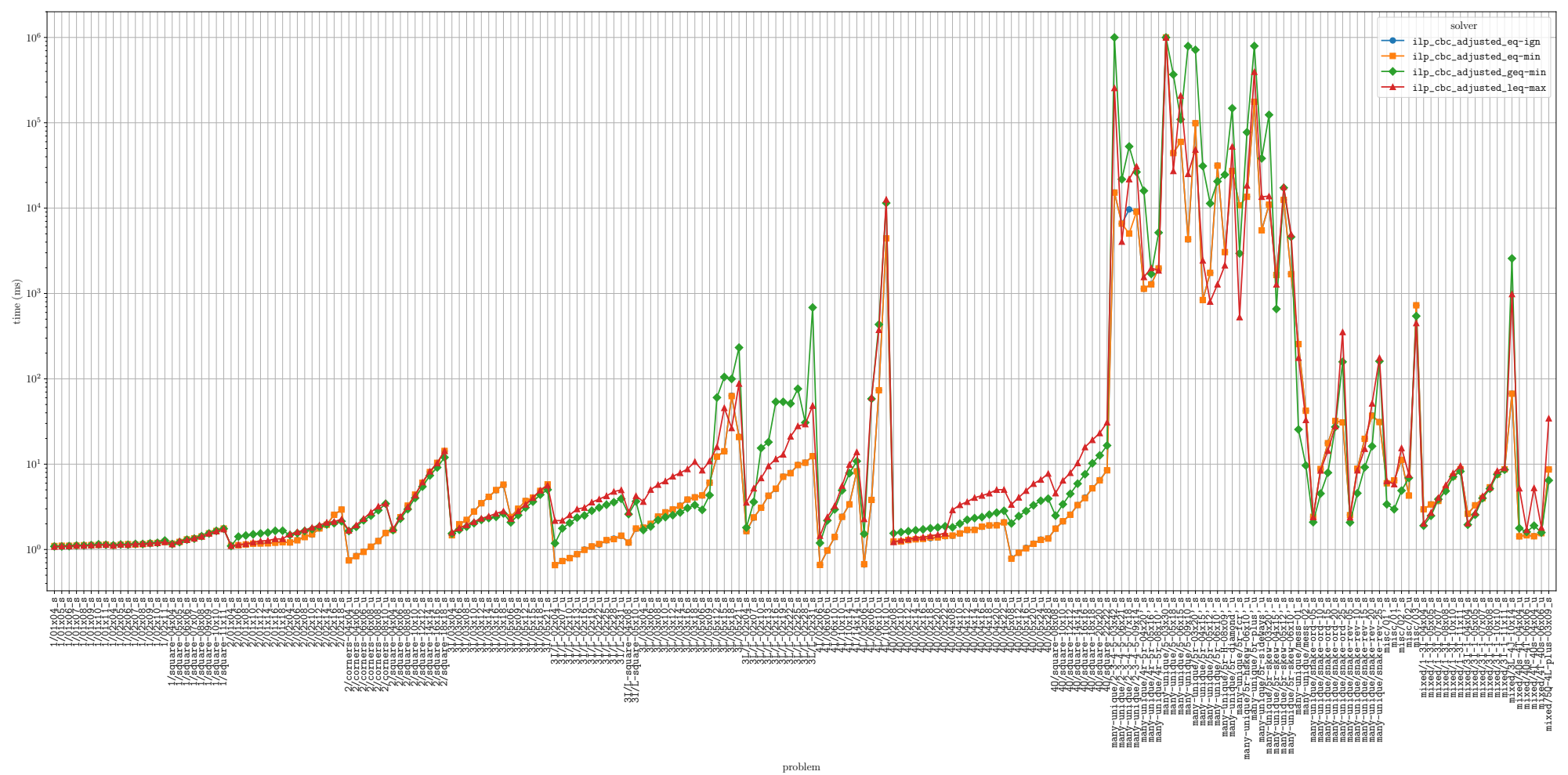
06-sat-amo-bimander



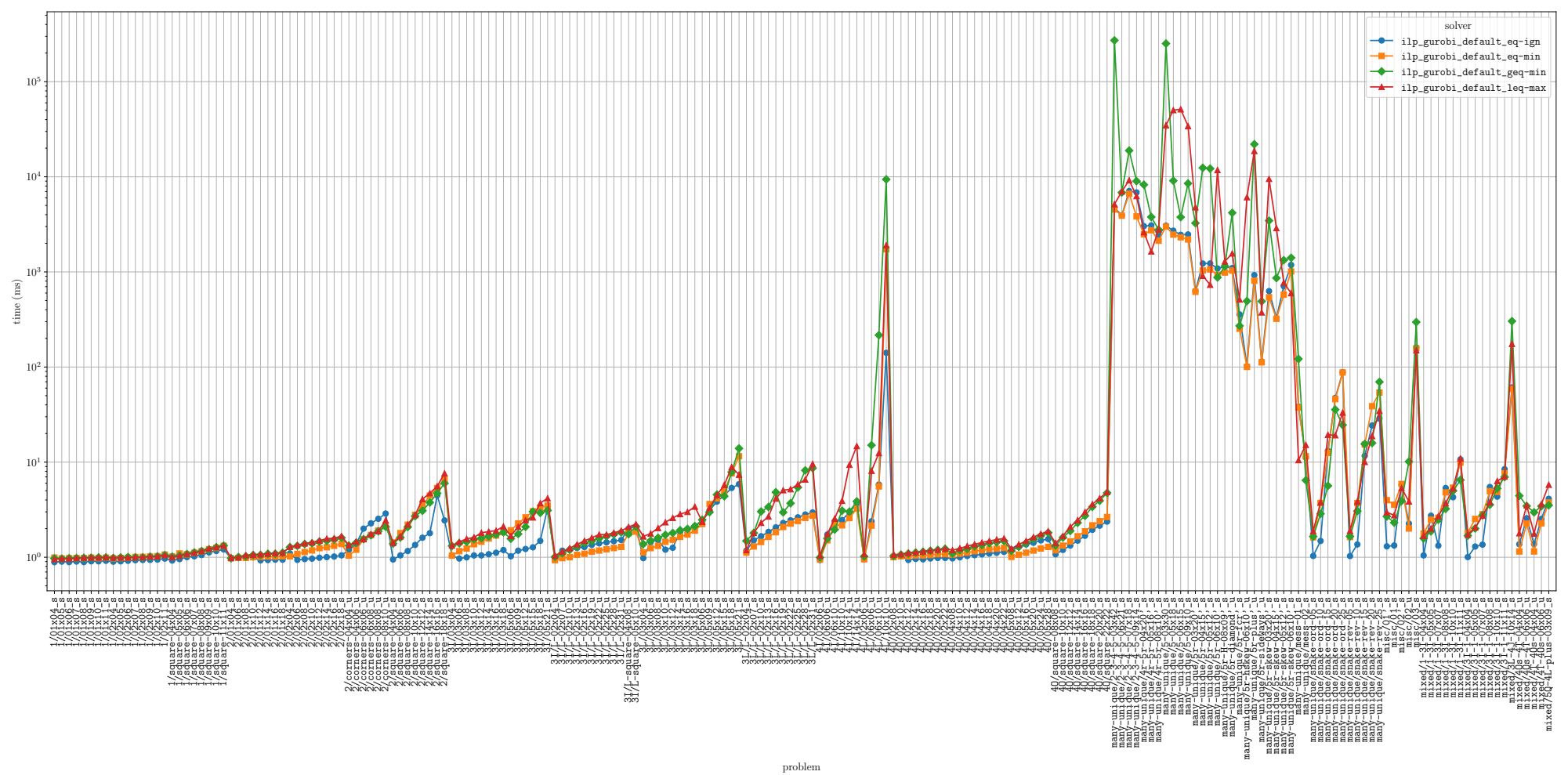
07-sat-amk-card



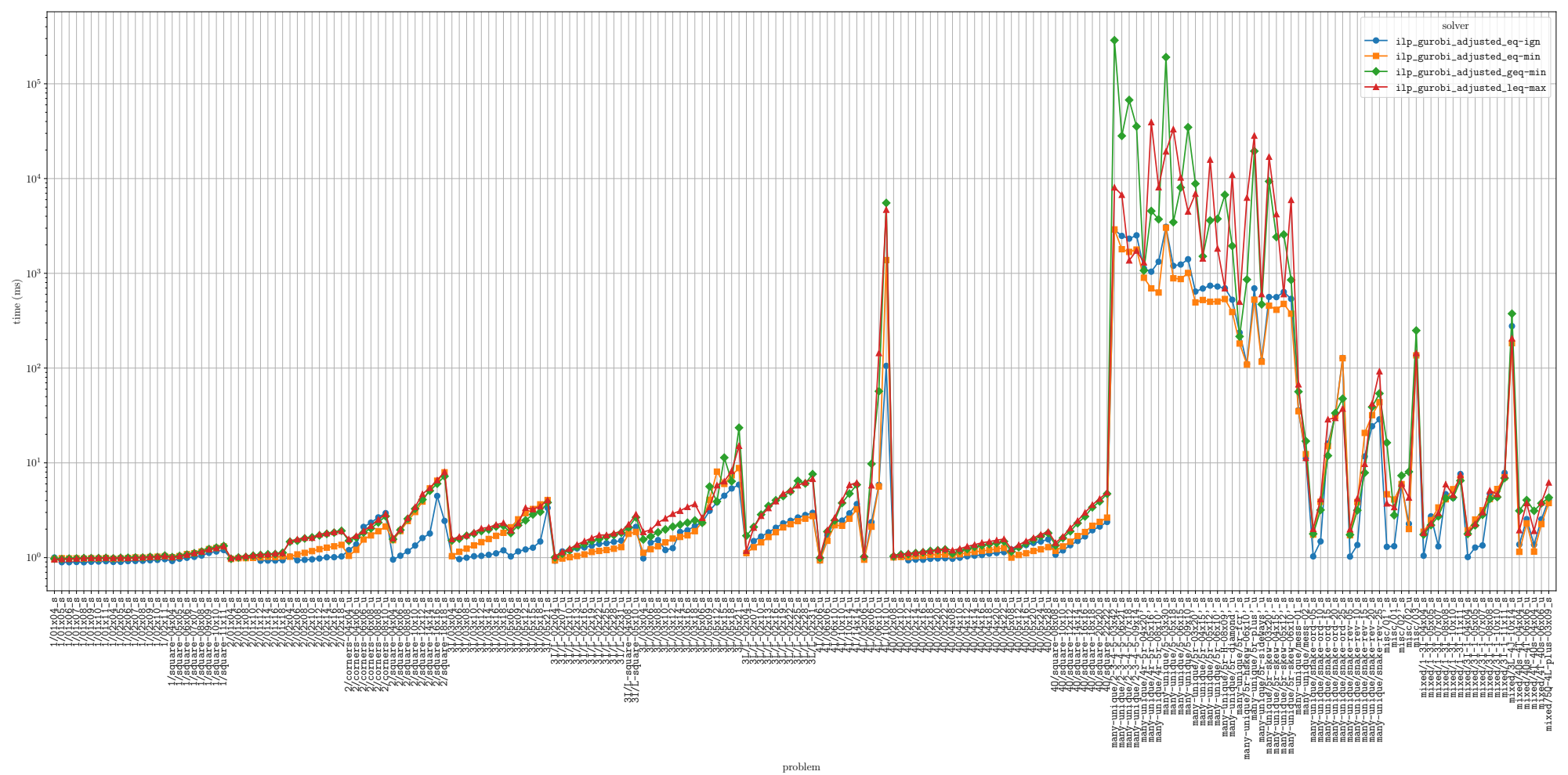
08-ilp-cbc-default



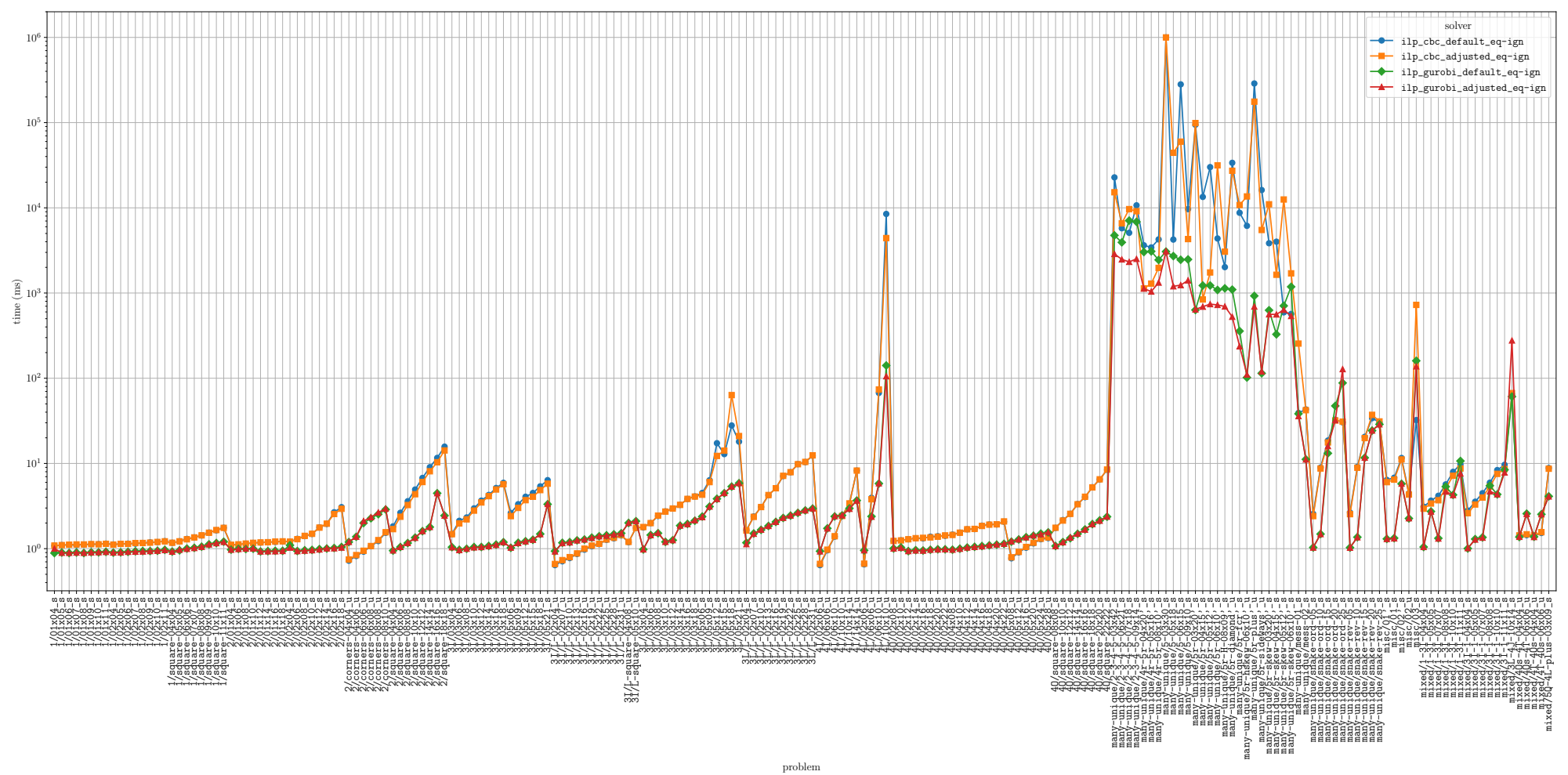
09-ilp-cbc-adjusted



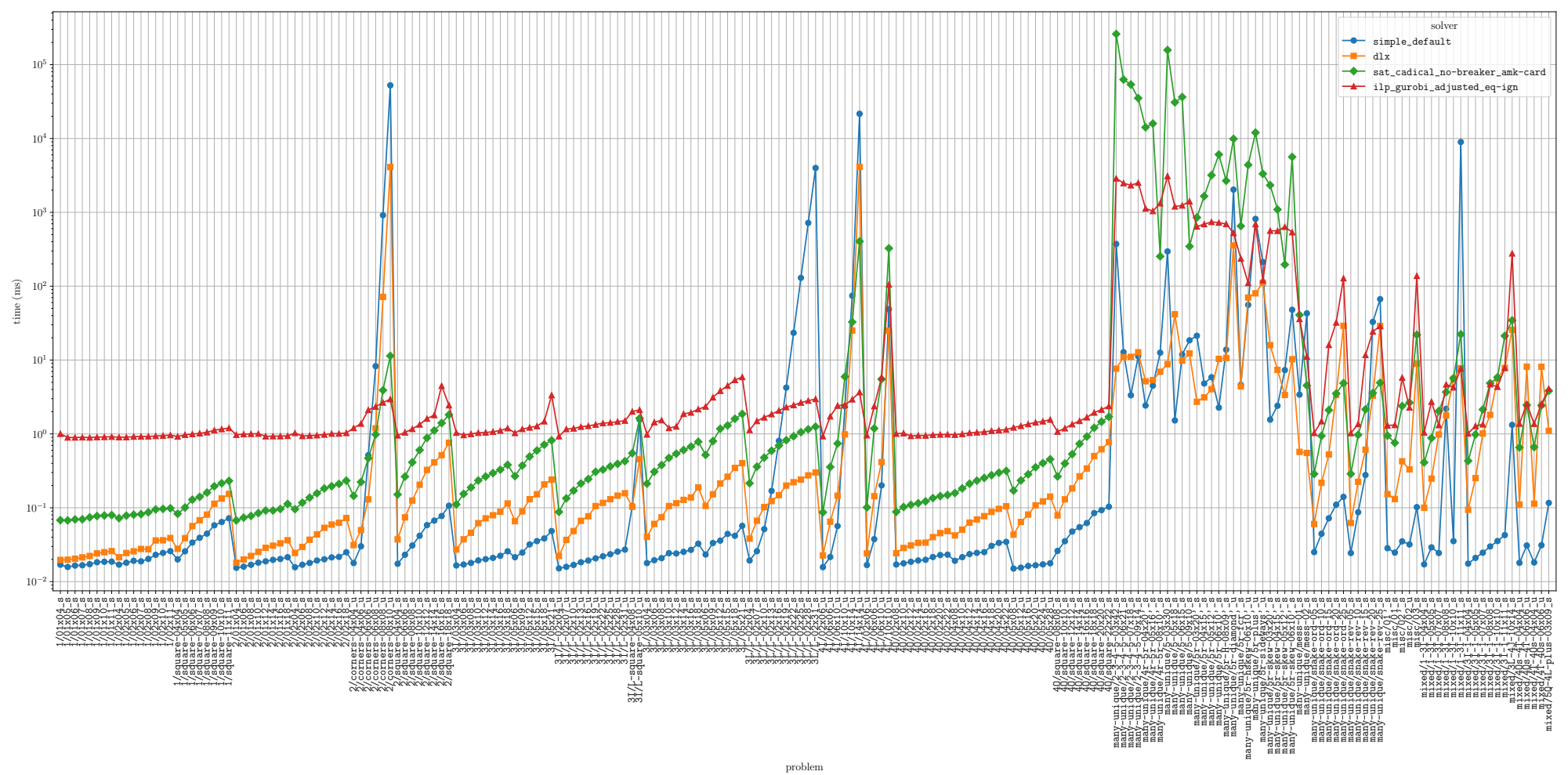
10-ilp-gurobi-default



11-ilp-gurobi-adjusted


solver
ilp_cbc_default_eq-ign
ilp_cbc_adjusted_eq-ign
ilp_gurobi_default_eq-ign
ilp_gurobi_adjusted_eq-ign
time (ms)
problem

12-ilp-eq-ign



13-representants