Experimentální porovnání algoritmů gSAT a probSAT

Luboš Zápotočný

České vysoké učení technické v Praze - fakulta informačních technologií
zapotlub@fit.cvut.cz

I. Úvod

Problém splnitelnosti booleovské formule (označováno z angličtiny SATISFIABILITY, zkráceně SAT) označuje problém nalezení splňujícího (vyhovujícího) ohodnocení logické formule v konjunktivní normálním formě tak, aby byly všechny její klauzule splněné.

SAT byl první problém o kterém se dokázalo, že je NP-úplný [1]. Tedy pro tento problém neexistuje (za předpokladu P $\neq$ NP) efektivní algoritmus, který by tento problém řešil v polynomiálním čase. Jelikož se jedná o NP-těžký (NP-úplné problémy jsou podmnožinou NP-těžkých) problém, lze na tento problém převést instance všech problému ze tříd P a NP.

Tato práce se zaměřuje na dva algoritmy pro hledání ohodnocení pomocí heuristik.

Prvním z porovnávaných algoritmů je gSAT, který v prvním kroce vygeneruje náhodné ohodnocení proměnných a v následujících iteracích převrací ohodocení některých proměnných tak, aby minimalizovat počet nesplněných klauzulí. Tento krok je opakován do doby než je nalezeno splňující ohodocení nebo program překročí nastavený limit iterací.

Druhý z algoritmů je znám pod názvem probSAT od Adrian Balint [2], který v prvním kroce také vygeneruje náhodné ohodnocení všech proměnných ale při výběru nesplněné klauzule a proměnné v dané klauzuli používá statistické rozdělení, aby vybral proměnnou, která statisticky přinese největší pravděpodobnost úspěchu.

II. Metody

V rámci experimentálního vyhodnocení úspěšnosti jednotlivých algoritmů byly algoritmy omezené na 500 možností převrácení hodnoty proměnné (flip). Efektivita algoritmů byla testována na splnitelných instancích SAT formulí ze sad

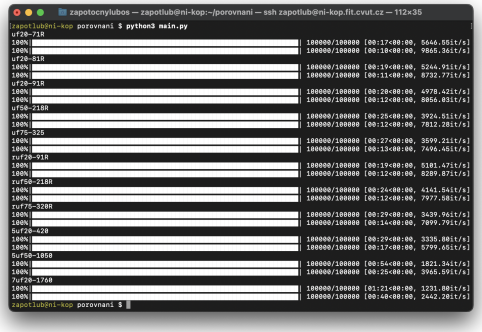
- uf20-71R
- uf20-81R
- uf20-91R
- uf50-218R
- uf75-325
- ruf20-91R
- ruf50-218R
- ruf75-320R
- 5uf20-420
- 5uf50-1050
- 7uf20-1760

Podrobný popis datových sad lze najít na stránkách kurzu NI-KOP ČVUT FIT [3]

Každá instance problému z datové sady byla spuštěna 1000 pro daný algoritmus, ale vždy s jiným seedem pro pseudo náhodný generátor. Ve výsledku byl každý algoritmus otestován na 100000 spuštěních pro každou datovou sadu. Celkově se tedy pro získání experimentálních dat spustilo 2.2 milionu běhů algoritmu pro hledání splnitelného ohodnocení.

Obrázek 1 zobrazuje dokočení sběru experimentálních dat na datových sadách pro algortmus gSAT (první řádek) a také probSAT (druhý řádek).

Algoritmy následně byly porovnány na základě úspěšnosti, tedy zdali daný alogorit-



Obrázek 1: *Spuštění experimentu*

mus dokázal vygenerovat splňující ohodnocení (splnitelné formule), a také na základě počtu iterací, než dané řešení dokázaly vygenerovat.

Zkoumán byl také poměr úspěšných a neúspěšných běhů algoritmu a také prúměrný počet iterací nutný k zastavení programu. Dále byl zkoumán průměrný počet iterací algoritmu pomocí váženého průměru, kde neúspěšné běhy byly penalizovány maximálním počtem iterací v dané datové sadě. Algorimy byly omezené na 500 iterací (flipů).

Následně byla zkoumána distribuční funkce počtu iterací v úspěšných bězích algoritmu a vzájemné křížení těchto distribučních funkcí mezi dvěma algoritmy. Také byla zkoumána korigovaná distribuční funkce, která zohledňuje neúspešné běhy algoritmu.

Nakonec bylo zkoumáno, zdali se počet iterací úspěšných běhů řídí log-normálním statistickým rozložením.

Tabulka 1: *Úspěšnost běhů*

Datová sada	gSAT	probSAT
uf20-71R	100%	100%
uf20-81R	99.9%	100%
uf20-91R	94.1%	98.2%
uf50-218R	49.7%	63.7%
uf75-325	27.4%	38.7%
ruf20-91R	94.0%	97.8%
ruf50-218R	51.7%	64.6%
ruf75-320R	30.0%	42.5%
5uf20-420	67.7%	79.5%
5uf50-1050	4.6%	5.4%
7uf20-1760	31.7%	37.6%
Průměr	59.2%	66.2%

Tabulka 2: *Průměrný počet iterací*

Datová sada	gSAT	probSAT
uf20-71R	8.0	8.0
uf20-81R	13.1	12.2
uf20-91R	104.1	76.9
uf50-218R	346.0	306.7
uf75-325	429.6	404.2
ruf20-91R	106.4	81.1
ruf50-218R	337.4	299.6
ruf75-320R	423.3	393.8
5uf20-420	277.8	233.7
5uf50-1050	489.3	488.9
7uf20-1760	412.6	396.5
Průměr	270.0	245.6

III. Výsledky

Tabulka 1 zobrazuje procentuální úspěšnost jednotlivých algoritmů na instancích z příslušné datové sady. Lze nahlédnout, že algoritmus probSAT byl na všech sadách úspěšnější (dokázal nalézt splnitelné ohodnocení proměnných). Algoritmus probSAT průměrně nalezl splňujicí ohodnocení v 66.2% případech, kdežto gSAT pouze v 59.2%, jedná se tedy o 7% rozdíl.

Tabulka 2 zobrazuje průměrný počet iterací, které algoitmus provedl, než našel splňující ohodnocení nebo byl po 500 iteracích zastaven a danou formuli nedokázal splnit. Lze také pozorovat trend, že algoritmus probSAT potřeboval průměrně méně iterací pro nalezení řešení než gSAT. Algoritmus gSAT potřeboval průměrně 270 iteraci, než nalezl splňující ohodnocení, kdežto probSAT potřeboval pouze 245.6. ProbSAT tedy potřeboval v průměru o 24.4 iterací méně, než gSAT.

Tabulka 3 zobrazuje vážený průměrný počet itarcí algoritmu na instancích z datových sad. Vážený průměr je zde počítán pomocí penal-

Tabulka 3: *Vážený průměrný počet iterací*

Datová sada	gSAT	probSAT
uf20-71R	8.0	8.0
uf20-81R	13.1	12.2
uf20-91R	133.7	85.9
uf50-218R	597.4	488.3
uf75-325	792.4	710.8
ruf20-91R	136.4	92.2
ruf50-218R	578.9	476.9
ruf75-320R	774.7	681.2
5uf20-420	439.4	336.0
5uf50-1050	966.4	962.1
7uf20-1760	754.1	708.3
Průměr	472.3	414.7

izace neúspěšných běhů alogitmu. Za předpokladu, že alogitmus nalezl splňující ohodnocení, do průměru se připočítá pouze počet iterací při daném běhu. Pokud ale algoritmus nedokázal nalézt splňující ohodnocení, do průměru se započítává počet iterací algoritmu společně s penalizací, která činí maximální počet iterací algoritmu přes všechny běhy nad instancemi z dané datové sady.

Dále v rámci experimentálního porovnání byla zkoumána distribuční funkce (její odhad) počtu kroků (iterací) algoritmů. Obrázek 2 a obrázek 3 znázorňují pravděpodobnosti, že daný algoritmus skončí do X kroků ($P[x <= X]$). Distribuční funkce byla počítána pouze z úspešných běhů obou algoritmů.

Algoritmus probSAT zvítězil v 7 z 11 datových sad. Algoritmus gSAT pouze v 4 z 11.

Korigovaná distribuční funkce je přenásobená distribuční funkce prametrem ρ. Parametr ρ je dynamicky počítán pro každou sadu instancí a reprezentuje podíl počtu úspěšných běhů k počtu všech spuštěných běhů. Korigovaná distribuční funkce již není distribuční funkcí nějakého statistického rozložení, ale i přesto může sloužit jako dobrý indikátor pro porovnání algoritmů. Například lze také sledovat trend dominance, křížení funkcí a porovnání absolutního vítěze na dané datové sadě.

Algoritmus probSAT zvítězil ve všech datových sadách. Algoritmus gSAT nezvítězil v žádné datové sadě.

Tento výsledek dobře koreluje s předchozím výsledkem o úspešnosti algoritmů, kde probSAT měl větší úspěšnost než gSAT.

Obrázky 6 a 7 zobrazují hustotu pravděpodobnosti počtu iterací v úspešných bězích obou algoritmů a pomocí aproximace parametrů lognormalního rozložení také zobrazuje hustotu tohoto rozložení s danými parametry.

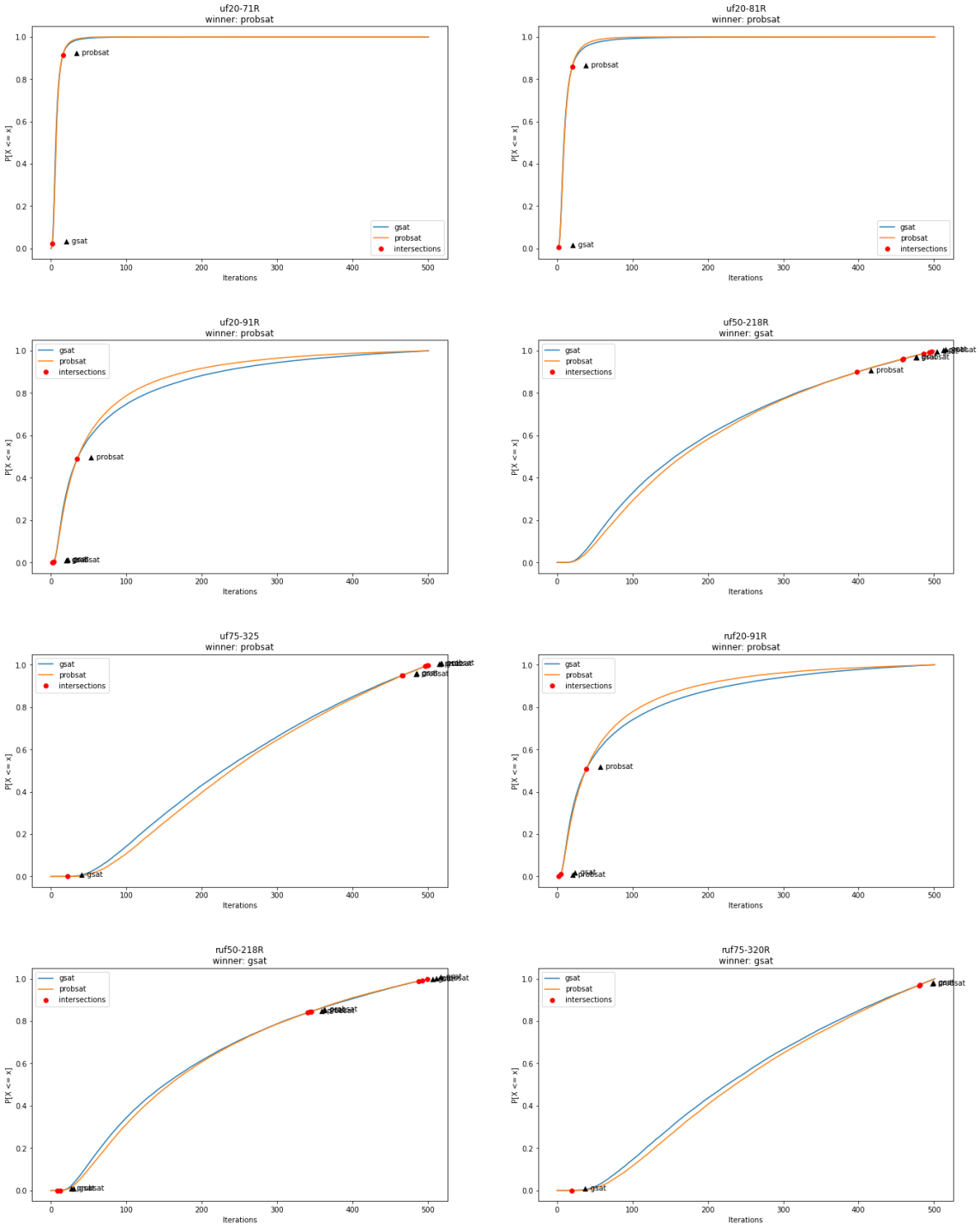
Zde je viditelné, že výsledné počty iterací lze na daných datových sadých považovat za lognormálně rozdělené.

IV. Diskuze

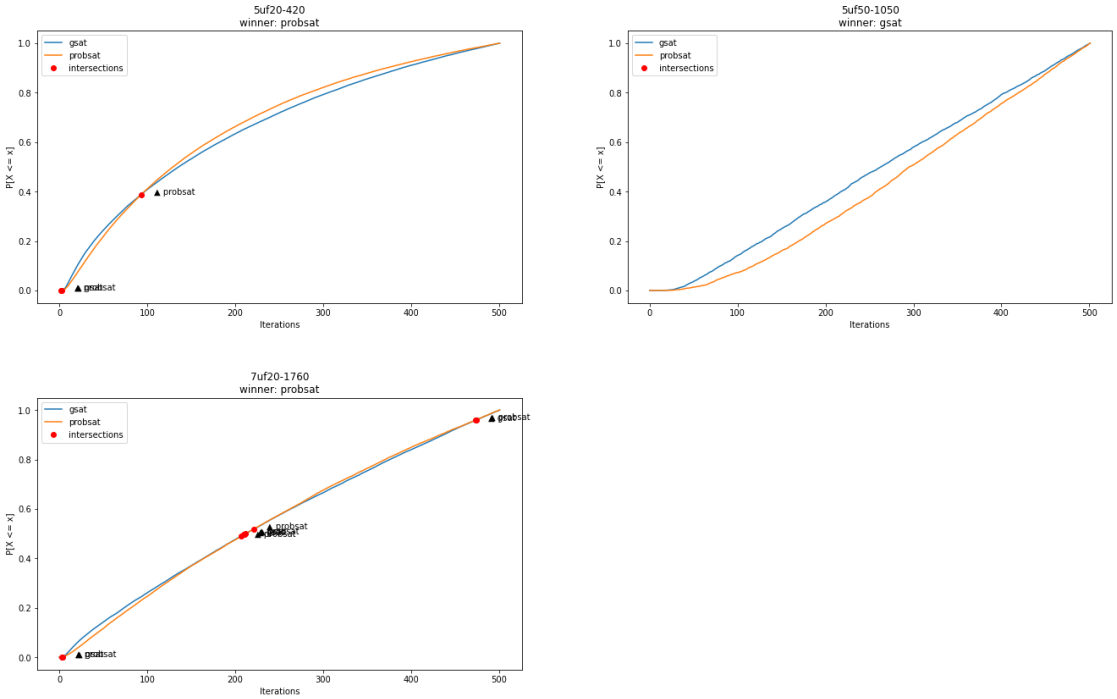
Na velkém výběru datových sad byly experimentálně porovnány dva algoritmy (gSAT a probSAT), které oba velmi dobře hledají splnitelné ohocnocení SAT formule. Z výsledků experimentu vyšlo, že probSAT má větší procentuální úspěšnost nalezení splňujícího ohodnocení se stejným limitem iterací jako gSAT. Také bylo pozorováno, že probSAT měl větší pravděpodobnost nalezení splňujícího řešení v daném počtu iterací.

Pro zajištění lepších výsldků a detailnějšího porovnání je možné použít další (větší) datové sady.

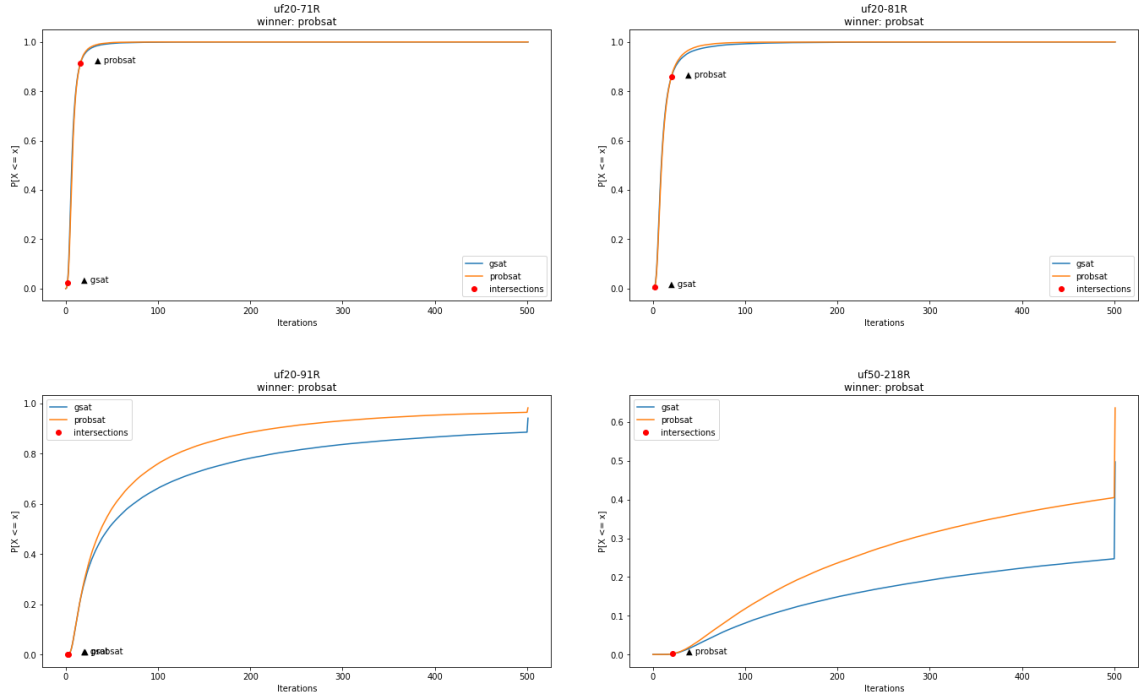
V průběhu tvorby experimentu vznikl nástroj na paralelní spouštění velkého množství datových sat pro programy gSAT a probSAT, který je možné nalézt na adrese github.com/zapotocnylubos/ni-kop-compare. Tento repozitář také obsahuje Python Notebook, který byl použit pro analýzu výsledků experimentu a generovaní grafů.



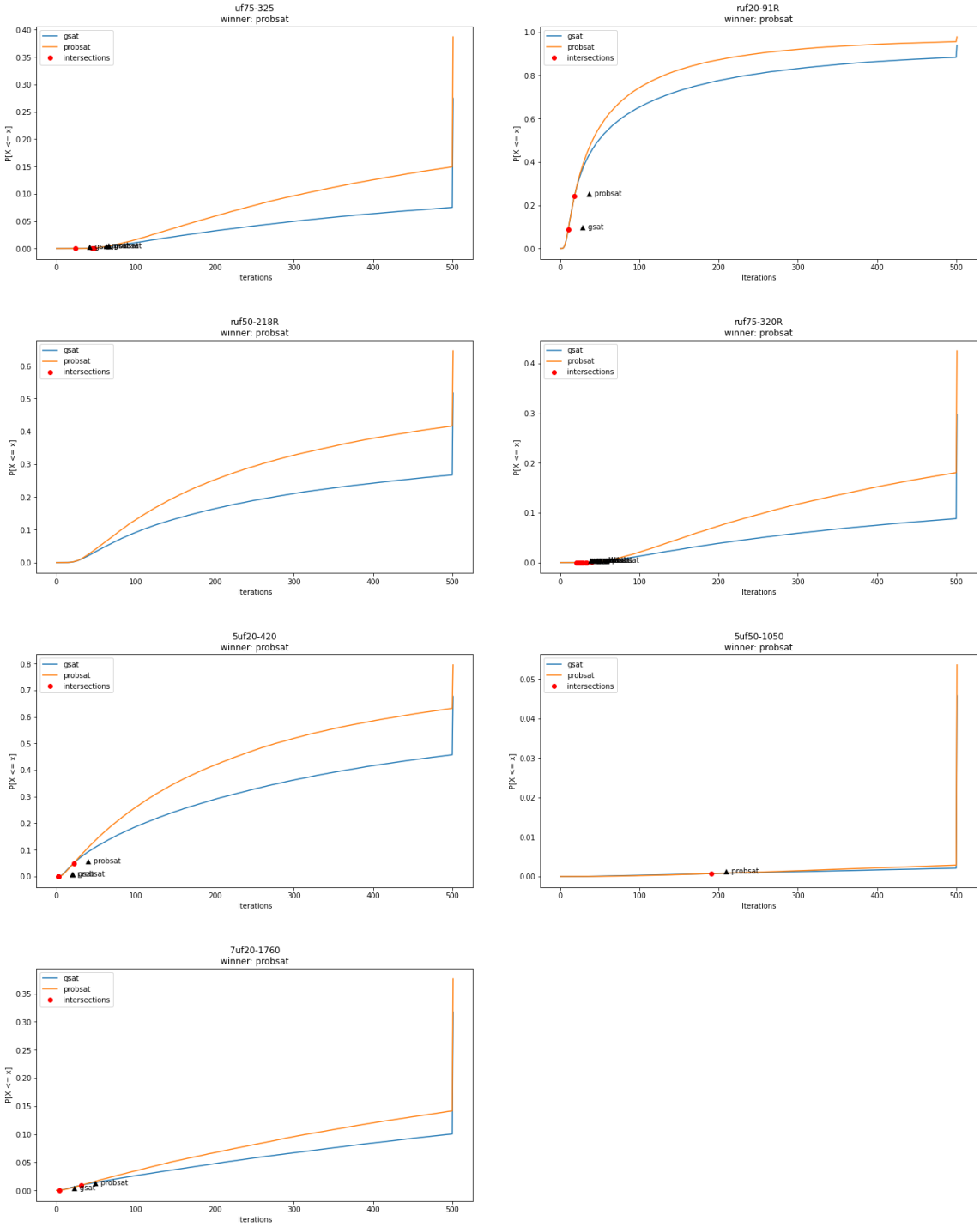
Obrázek 2: *CDF pro úspěšné běhy algoritmů*



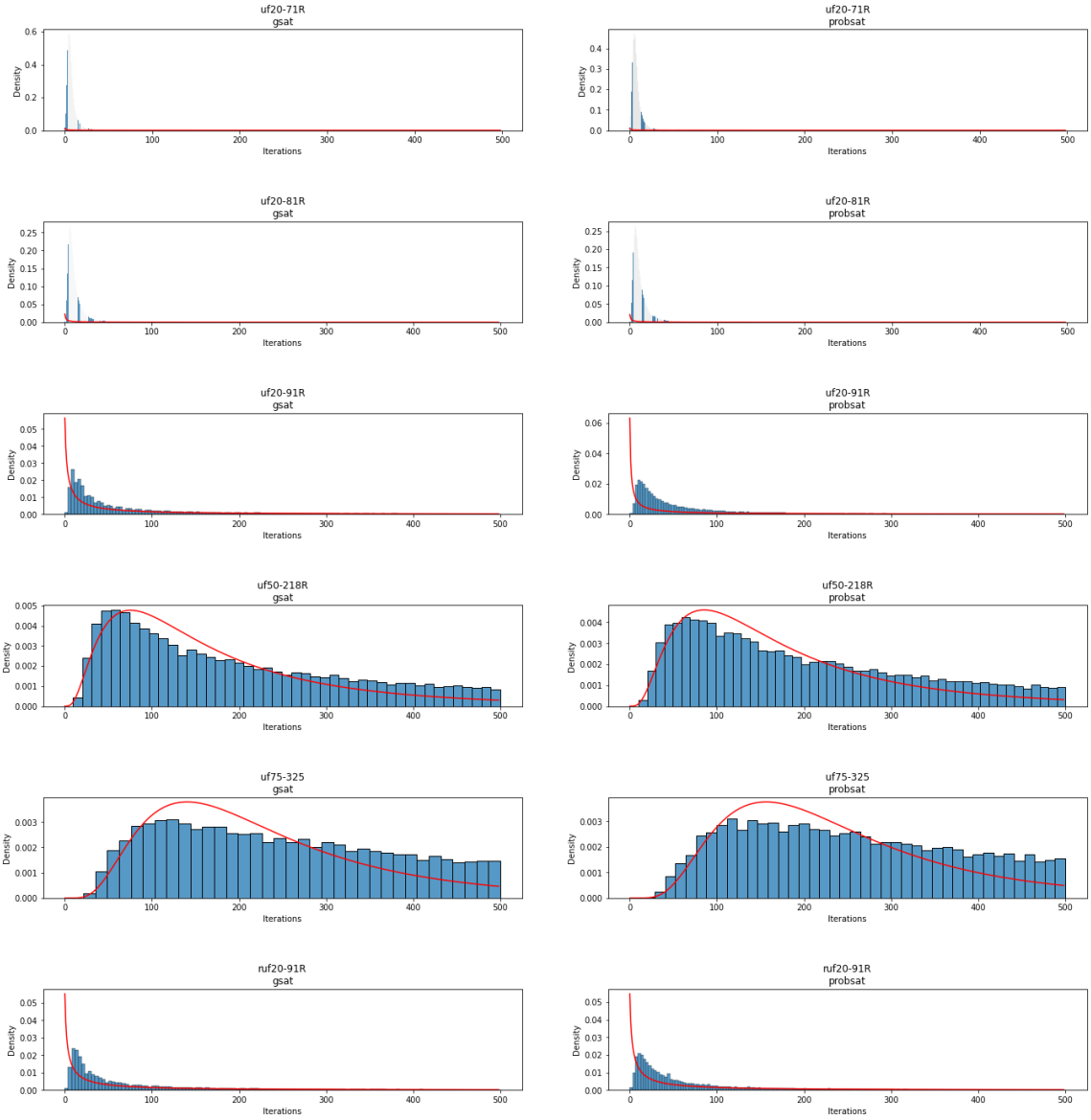
Obrázek 3: *CDF pro úspěšné běhy algoritmů*



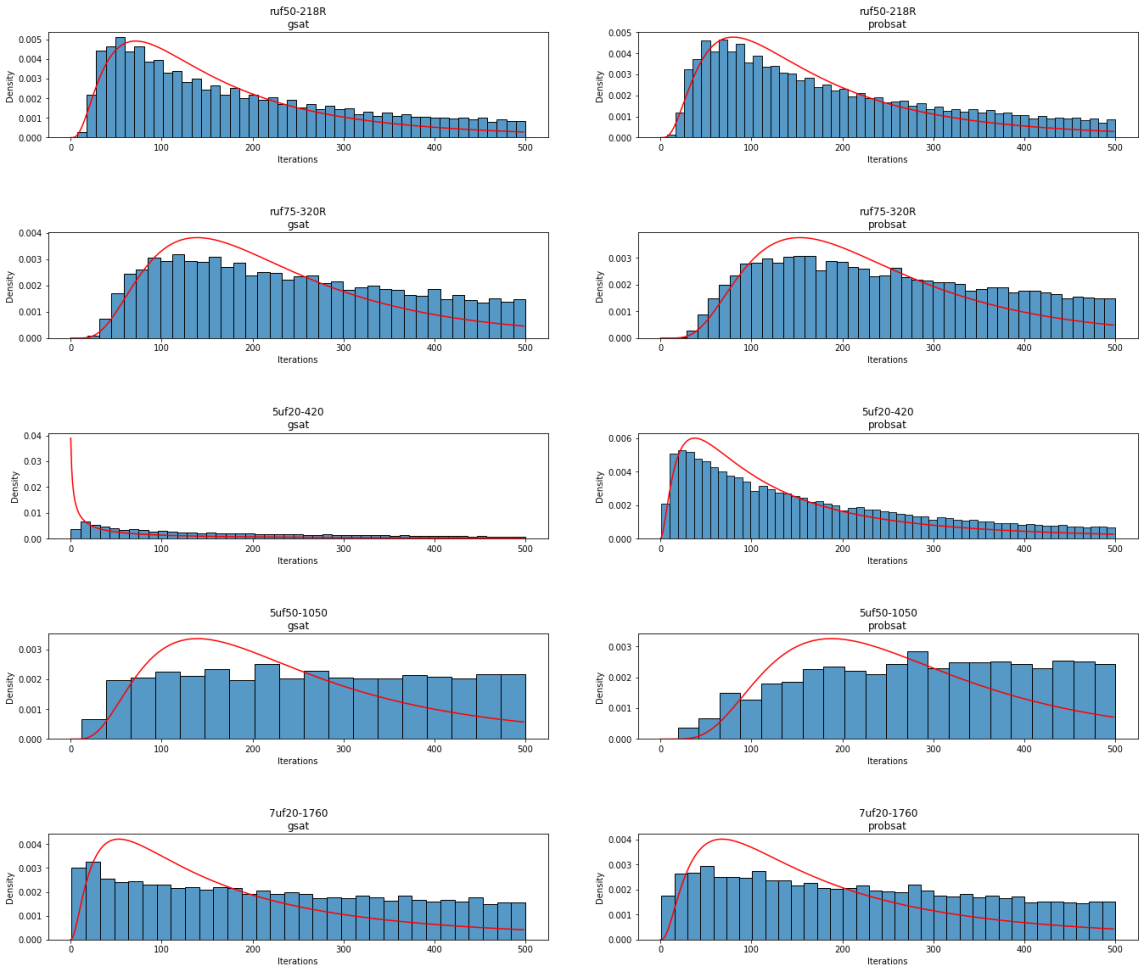
Obrázek 4: *Korigovaná CDF pro běhy algoritmů*



Obrázek 5: *Korigovaná CDF pro běhy algoritmů*



Obrázek 6: *Lognormální rozložení a hustota experimentálních dat*



Obrázek 7: *Lognormální rozložení a hustota experimentálních dat*

References

[1] Stoc '71: Proceedings of the third annual acm symposium on theory of computing. 1971.

[2] Adrian Balint and Uwe Schöning. Choosing probability distributions for stochastic local search and the role of make versus break. pages 16–29, 2012.

[3] Jan Schmidt. Data a programy (nikop). `https://courses.fit.cvut.cz/NI-KOP/download/index.html`.