

硕 士 研 究 生 读 书 报 告



题目 SATenstein:从元部件中自动构

建本地搜索求解器

作者姓名 王晓红

作者学号 21551073

指导教师 李启雷

学科专业 移动互联网与游戏开发

所在学院 软件学院

提交日期 二○ 一 五 年 12 月

SATenstein: Automatically Building Local Search SAT Solvers

From Components

Ashiqur R. KhudaBukhsh, Lin Xu, Holger H. Hoos, Kevin Leyton-Brown

Department of Computer Science

University of British Columbia

{ashiqur, xulin730, hoos, kevinlb}@cs.ubc.ca

**摘要**

设计一个高性能的解决复杂问题的计算是困难且耗时的。在这种情况下，我们希望能在本地随机搜索（SLS）时这项工作可以自动执行且满足要求。 文章首先介绍它的发展史以及高参数化的解决框架、我们称之为SATenstein。 它包括我们收集或者从已存在的高性能的SLS（自动化本地搜索）SAT的组件中受到的启发。SATenstein的参数控制了应用在特殊实例和这些实例的行为的组件的选择.基于SAT的解算器，SATenstein可以被配置到广泛的现有高性能的SLS（本地随机搜索）并且算法还十分新颖，我们在使用自动算法配置的过程中发现几位大名鼎鼎且具有挑战性的分布表现良好的SATenstein的实例。总而言之，我们最终用最少的人工工作量得到了优于当前为止表现最好的的SLS（自动化本地搜索）算法

关键字：SATenstein，SLS(本地随机搜索)

**Abstract**

Designing high-performance algorithms for computationally hard problems is a difficult and often time-consuming task. In this work, we demonstrate that this task can be automated in the context of stochastic local search (SLS) solvers for the propositional satisfiability problem (SAT). We first introduce a generalised, highly param- eterised solver framework, dubbed SATenstein, that in- cludes components gleaned from or inspired by existing high-performance SLS algorithms for SAT. The param- eters of SATenstein control the selection of components used in any specific instantiation and the behaviour of these components. SATenstein can be configured to in- stantiate a broad range of existing high-performance SLS- based SAT solvers, and also billions of novel algorithms. We used an automated algorithm configuration procedure to find instantiations of SATenstein that perform well on several well-known, challenging distributions of SAT in- stances. Overall, we consistently obtained significant improvements over the previously best-performing SLS algorithms, despite expending minimal manual effort.1

**1介绍：**

一位辉煌和痴迷的科学家—Victor Franken- stein，提出了以创建清理身体某部位“完美”的人。我们基于相同的总体思路来开展这项工作：我们建立利用现有的高性能求解器清理对给定问题的组件的新算法。与科学怪人不同的是，我们使用一个自动化的建设过程，这使我们能够建立算法，其性能是针对特定类型的任务（即问题的实例），以最小的人的努力进行了优化。

在传统的方法来构建启发式算法，大多数设计选择被固定在开发时，提供少量的参数给用户。与此相反，我们主张提供尽量少的参数给用户，取而代之的是将大量的设计参数在运行时进行配置。这免去了设计者做出决定初期不知道他们将如何与感兴趣的实例分布等算法组件进行交互的负担。相反，反倒像是有前途的，在已知的算法元素以及新机制绘图设计人员可以包括许多其他方法来解决同一子问题。当然，要解决实际问题，这样的框架必须作出通过设置其参数来实例特定算法。我们用一个黑盒子算法配置过程作出这种选择感兴趣的实例分布。

当然，我们不是第一个使用自动化的方法来搜索大设计空间来提出建设算法的。与此相反，我们的工作可以看作是一个一般性和不断增长的趋势的一部分，助长了用于高性能求解器在实际应用中难以组合问题的需求日益增加，由希望减少需要用于建立这种算法的人的努力和较低的计算能力，可以被利用用于自动化算法设计过程的部件的不断增加的可用性，有沿着这些路线我们讨论了在第2节最密切相关的工作细节。

在广泛的尝试中，我们的做法主要区别在三个方面。首先，它准确的从高性能算法中区分出来巨大的组合设计空间。第二，它使后面的阶段完全使用自动化进程。这使得它可以用基本的计算能力而不是人力来为各种各样类型的输入获得特殊的解算器。命题满足性问题（SAT）：最后，它是由一个事实，即我们得到先进国家的新的求解器在计算机科学中最具挑战性和广泛的研究问题之一验证。我们还注意到，我们的方法不同于自动算法选择方法，如SATzilla[Xu等人，2008]，它选择的几个求解一个要在根据该实例的特性的给定问题的实例运行。事实上，这两种方法是互补的：像SATzilla方法可以使用自动设计方法开展这项工作得到求解器的优势。

虽然我们已经概述的方法并不限定于特定的域，本文重点研究的随机局部搜索（SLS）算法的SAT自动化施工。这是是一个具有挑战性的自动化算法设计，因为一个广泛而持续的研究工作已经进入设计阶段来设计高性能的SLS算法自上世纪80年代后期（手动）。 SLS基于解算器继续代表求解各种类型的SAT实例的技术状态。他们在SAT算法的自动选择中也起着重要的作用。

我们通过实验证明，我们的新的，自动构造求解跑赢最好的SLS为基础的SAT求解器目前可通过六个显著保证众所周知的SAT实例分布，

从硬随机3 - SAT实例SAT编码的保理和软件验证的问题。因为SLS基于SAT解算器是为解决4这六个基准分布的最佳已知方法，我们的新解算器表示在本领域用于解决各个子类的SAT的状态的重大进展。在剩余的两种类型的实例，特别是关于SAT编码的软件验证的问题，我们的新解算器缩小的最佳SLS算法的性能和最佳的基于DPLL解算器之间的间隙。

我们证明通过实验，我们的新的，自动构造解算器优于目前可通过六个显著保证金众所周知的SAT实例分布的最好的SLS为基础的SAT求解器。

从随机3 - SAT实例SAT编码的保理和软件验证的问题。因为SLS基于SAT解算器是为解决4这六个基准分布的最佳已知方法，我们的新解算器表示在本领域用于解决各个子类的SAT的状态的重大进展。在剩余的两种类型的实例，特别是关于SAT编码的软件验证的问题，我们的新解算器缩小的最佳SLS算法的性能和最佳的DPLL基于解算器之间的间隙。

2 相关工作

值得注意的是，Fukunaga自动构建了一个通用的SLS算法SAT变量选择机制的遗传编程[Fukunaga, 2002; 2004].  该设计空间他的做法被认为可能是无界的，并且他的基因编程程序量身定制，以搜索这个空间。直到2000年，GSAT-基础和WalkSAT型SLS算法和候选变量选择机制为唯一评价随机-3-SAT实例至多100个变量。与此相反，我们考虑的基础上，从25的最好的SLS算法的SAT目前可用的拍摄部件算法巨大的，但有界组合空间，我们使用一个现成的，货架，通用算法配置过程，以搜寻该空间。在此期间的元算法搜索过程中，候选解算器，是与多达4 878变量和56 238的条款的SAT实例进行评估。通过Fukunaga的所获得的最好的自动构建的求解实现类似于在中等规模的SAT实例在2000中提供的最佳WalkSAT变体的性能水平，但并没有持续改善的最佳SLS基于SAT算法当时的性能。相比之下，我们的新SATenstein求解器进行大大超过租金国家的最先进的电流SLS为基础的SAT求解器更好地在各种具有挑战性的SAT实例。最后，虽然Fukunaga的的原则进近可以用来获得高性能的求解器对特定类型的SAT实例，就我们所知，这潜能还未实现。我们的方法，在另一方面，是专门用于自动构建给定实例的分布设计的高性能求解器，我们的经验结果清楚地表明，它在实践中工作得很好。

其他两件作品来自我们自己的小组在概念上与这里介绍的工作。Hutte等。 [2007年a]使用的自动算法配置过程细化高度参数DPLL型SAT解算器，枪（随后将其调谐）使用相同的配置过程中，为了解决SAT-编码的硬件和软件验证问题，分别。这些自动调谐矛版本进行了论证，以改善在时刻解决这些类型的SAT实例的现有技术的状态。最近使用过的同一个自动化的算法，配置过程，设计模块化SLS算法时间表问题，随后放在第三的2007年国际排课比赛的类别之一。在这两种情况下，自动配置过程被用作人性化的设计过程中的工具，而在这里提出的工作提供了基础此处理后的不同的和完全自动化的阶段。

另一种概念上的关系存在于我们自己的工作SATzilla这两种方法都使用自动方法从一组算法选择，以实现对SAT实例给出分布改进的性能。 （而且，当然，这两个具有类似的名称！）然而，SATzilla执行从一组固定的黑盒SAT求解每个实例的选择，每个解算器依赖于特定实例的特性和大量的运行时数据。与此相反，SATenstein是从组件的方法自动建立新的求解器。同时，这些组件定义候选算法，其中大部分以前从未深入研究。我们主张实例求解器从该空间仅考虑分布的基础并不考虑个别问题的实例的特征。此外，该搜索这个空间算法配置工具仅限于收集运行数据，对于大多数配置算是非常有限。事实上，我们的两种方法是高度互补的：SATenstein可以用SATzilla来获得新的SAT的算法。事实上，SATzilla的最新版本（最近提交给2009年SAT竞赛）使得使用SATenstein使用多个求解器构造[徐等人，2009]。

在算法综合现有的工作主要集中在满足给定的正式规范，或从一个大的且多样化的领域（例如，见[Westfold和史密斯，2001年解决特定问题自动生成算法; Monette等，2009; Gaspero和Schaerf，2007]，而我们的工作主要集中在性能优化中，它们都保证是通过建设正确的候选人求解空间大。显然，这两种方法之间的互补性由于侧重点的不同而被认为是在算法合成和面向性能自动化算法的设计有很大的不同

最后，我们的工作从根本上建立在大量的研究是引起了高性能从中SAT求解器，我们采取了组件我们SATenstein求解器（见第3节），并在UBCSAT的解算器框架SLS为主SAT算法[汤普金斯和胡斯，2004年]，在其顶部我们实施SATenstein。

**3 SATenstein-LS**

几乎所有现有的基于SLS SAT解算器可分成四大类：基于GSAT，基于WalkSAT，基于动态本地搜索和G2WSAT算法。虽然基于GSAT的算法具有相当的历史意义，国家的最先进的无本SAT算法属于这一类。因此，我们构建了我们的解算器框架，以从剩余的三个家庭跨越高性能局部搜索算法。

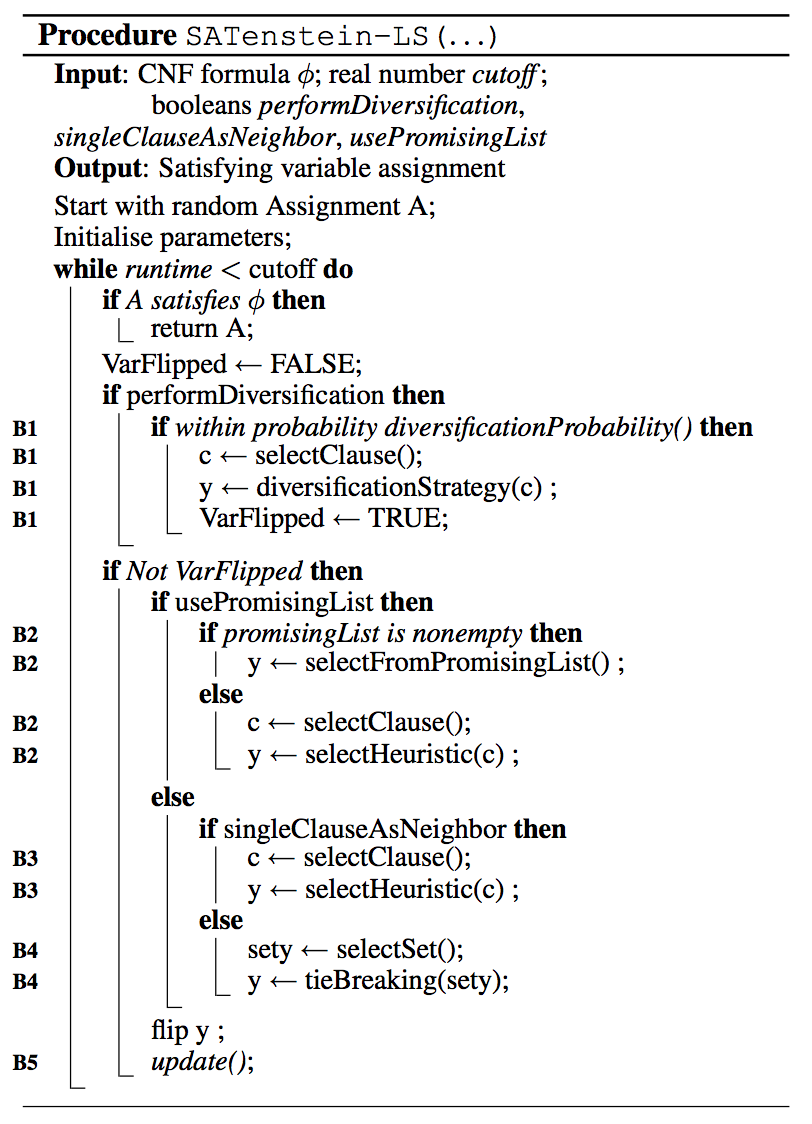
这个框架被描述为程序SATenstein-LS。伪代码大致分为五个部分，记B1-B5。一个SATenstein-LS实例具有以下一般结构：

1.选择性的执行B1来搜索的多样化[范和格雷顿，2007;胡斯，2002;

2.获取变量通过执行B2，B3中的一个翻转，和B4-从而确定是否是一个G2WSAT的衍生（B2）、基于WalkSAT（B3）、或动态局部搜索算法（B4）的实例，然后翻转这个变量;

3.选择性的执行B5进行更新到变量列表条款处罚或动态调整算法参数。

**伪代码：**



我们每一个代码块是由一个或多个部件组成的，其中一些是在多个块共享建造。这些成分归纳在表1中各成分依次配置的，即一个或多个参数（41总共;并非所有这些都反映在高级别伪代码），从高性能SLS算法不同的核心的启发式之间进行选择，这些是我们暴露在命令行和调整使用我们的自动配置参数。因为有些参数有条件地依赖别人，所以很难确定有效的SATenstein-LS实例的确切数字;一个保守的下限为2×（10的11次）

现在，我们给每一个构建块的一个高层次的描述。B1取决于selectClause()，diversificationStrategy()，和diversificationProbability()分量。组件selectClause()有一个明确的参数作为输入并根据其值，选择一个假子句一致地随机或概率正比于它的条款罚[Tomp-金穗和胡斯，2004]。组件diversificationStrategy（）可以通过一个明确的参数来配置做任何的概率diversificationProbability（）：翻转最近最少翻转变量[李和黄，2005年]，翻转最不经常翻转变量[普雷斯特维奇，2005年]，翻转可变最小变权[普雷斯特维奇，2005年]，或翻转随机选择的变量[胡斯，2002]。

表1：*SATenstein-LS* components.

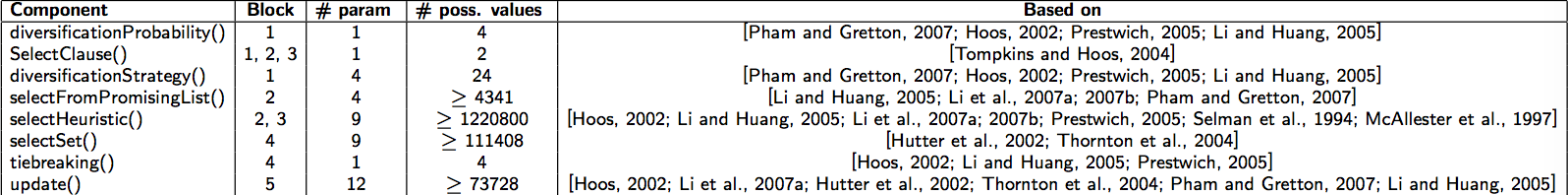
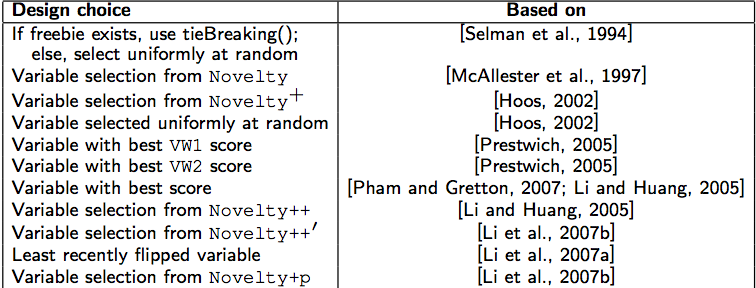


表2：Design choices for selectFromPromisingList().



代码块B2也会根据该G2WSAT体系结构求解器，它使用一个“有前途的列表”跟踪一组被翻转的变量。两种策略，用于选择最终将要从前景的列表翻转变量已经在文献中提出：选择具有最高分数[Li和黄，2005]，并选择最近最少翻转变量的变量。我们增加了9个新的选择基于其他求解启发式，据我们所知，以前从来没有被应用到G2WSAT-变种算法。 （例如，我们相信，从新颖变体的可变选择机制仅被应用到不满意条款，而不是期望的列表。）在表2中给出11个selectFromPromisingList（）的每个启发式可能的值

如果期望列表是空的，代码块B2的行为与代码块B3相似，它实行WalkSAT的算法。首先一个条款，C是使用B1已经描述的selectClause（）分量选择。然后把变量翻转从c。使用selectHeuristic（）成分中。该组件可以被配置成实现13个不同的变量能够选择策略，包括那些来自WalkSAT/ SKC[塞尔曼等人，1994]，VW1和VW2[Prestwich镇，2005]，和VAR-欠条新颖变体。在gNovelty+选择策略还包括一个可选的“平举”机制;我们扩展了该可选机制，为所有其他新奇变种的选择策略。

B4座动态实例化局部搜索算法。所述selectSet（）组分认为该组中出现的任何不满意子句变量。它与关联每一个这样的变量VA得分，这取决于每个当V被翻转，改变满足性地位条款的条款权重。这些条款的权重反映卫星 - isfying每个子句的感知重要性;例如，权重可能会增加更长的条款已经满足，并降低事后[胡特等人，2002; Thornton等人，2004]。进球后的变量，selectSet（）返回与要得高分所有变量。我们实施这个组件包括三个不同的计分函数，包括那些因McAllester等人。 [1997]，塞尔曼等。 [1994]，和一种新型的，贪婪变量蚂蚁只考虑的是由一个可变倒装满足先前不满意子句的数目。的难解难分（）组件选择从最大得分根据所使用的diversification-战略（）成分相同的策略，设置一个变量。

代码块B5更新数据结构底层后一个变量已经被翻转。以有效的方式执行这些更新是在优化SLS算法的性能的一个核心问题;在与该SATenstein-LS框架支持的机制从许多不同的SLS算法的组合中，每个视不同的数据结构的事实相结合，这呈现的向上日期（）技术上相当有挑战性的实施。

我们通过仔细检查自己的行为当配置为实例10突出的算法（参见[KhudaBukhsh，2009]了解详细信息），相对于独立的参考实现验证SATenstein-LS的正确性。我们也比较了不同的实现中的经验性能方面，发现在某些情况下，它们实际上是相同的（用于SAPS，RSAPS，和adaptNovelty+），而在另一些SATenstein-LS是慢（最差为2.17倍慢，对于gNovelty+）。主要的原因SATenstein-LS和参考实现之间观察到的性能差异是在更新（）成分缺乏专用数据结构的优化。

**4实验装置**

我们认为六套为SAT公知的基准实例，在表3中列出的由于SLS算法无法证明不可满足，我们构建我们的基准设置为只包括满足的实例。该HGEN和FAC发行只包括满足的情况下，对于这些分布，我们产生2000实例，并划分这些随机分为含每1000实例的训练和测试集。对于其余的分布，我们筛选出使用完整的解决者不可满足的情况下。对于QCP，我们产生23 000实例周围的溶解度相跳变，使用由戈麦斯和塞尔曼[1997]给出的参数。我们使用的是完整的求解器解决了这些，然后随机选择了2000满足的情况下。这些我们随机分为每1000个实例的训练和测试集。对于SW-GCP，我们产生20 000实例[绅士等人，1999采用了完整的解算器，以随机抽样2000满足的情况下，我们又随机分为每1000个实例的训练和测试集。对于R3SAT，我们产生了一组1000个实例有600个变量，4.26一条款对变量的比例。我们确定了使用完整的求解器521可满足的实例; 3我们随机选择这些实例250，以形成一个训练集和250，以形成一个测试组。最后，我们使用了CBMC发生器基于与不同阵列的尺寸和环展开值的二进制搜索算法来创建611 SAT-编码软件验证实例。由此获得的情况下，我们筛选了七项不可满足的情况下，并确认所有其他的满足的。我们随机分为剩余的604实例到303实例的训练集和测试集301的实例。

为了对SATenstein-LS框架执行自动算法配置，我们首先使用一个目标函数来量化性能;我们选择了把重点放在平均运行时间。然而，高效的算法配置器之三minate（或上限），他们之前的一些运行完成，使得平均不明确。因此，下列胡特等。 [2007年a]，我们定义一组N的惩罚平均运行时间（PAR）运行，用K-第二断为平均运行时间在所有N运行，以算作10·K秒上限运行。

由于我们的算法配置，我们选择了FocusedILS程序从ParamILS框架2.2版，因为尽我们所知，这是能够在非常大的，离散的参数空间有效地工作的唯一方法。我们令k到五秒钟，并配发两天FocusedILS的每次运行。因为佛cusedILS是一个随机过程，其性能可在很大多个独立运行显著，特别是DE-未决它选择为在训练集中的实例的数量级。我们跑了10次每次训练集，使用不同的，随机确定的实例排序每次运行。从FocusedILS每个实例分布D中得到的10参数配置，我们选择与在训练集的最佳惩罚平均运行参数配置，我们指SATenstein-LS的相应实例作为SATenstein-LS[D]。

对于每一个分布D，我们比较SATenstein-LS [D]。对了11个高性能SLS型SAT的测试集求解器的性能。我们包括每一个SLS算法，在过去的五年里[SAT比赛，2009年]赢得了金牌，在任何种类的SAT竞争，也包括其他一些著名的，高性能的算法。这些列于表4。

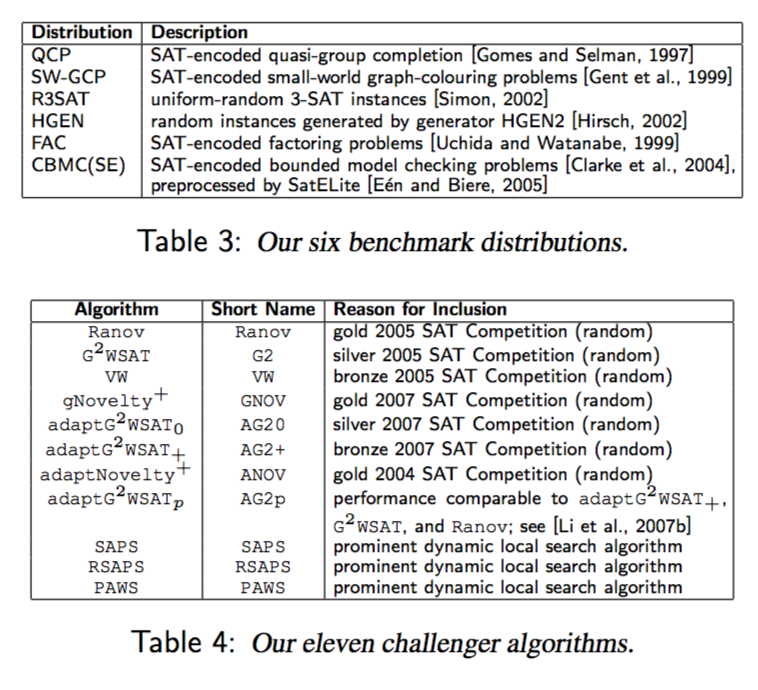
SATenstein-LS可以实例化模拟所有11的这些挑战者算法，所不同的是它不支持所使用的Ranov，G2和AG20预处理部件。所有我们的实验比较是基于原始算法的实现中，提交给相应的SAT比赛，除了PAWS，的量，UBCSAT执行几乎与在运行方面的原始解算器。所有比较是基于与每运行600秒的截止时间运行的每个解算器的10倍。

我们进行了我们的实验中55双3.2GHz的英特尔至强电脑具备2MB缓存和2GB RAM，运行的OpenSuSE 10.1的Linux集群上;运行时间为所有的算法（包括FocusedILS）进行测量CPU时间对这些参考的机器。

对于每一个分布D，我们比较SATenstein-LS [D]。对了11个高性能SLS型SAT的测试集求解器的性能。我们包括每一个SLS算法，在过去的五年里[SAT比赛，2009年]赢得了金牌，在任何种类的SAT竞争，也包括其他一些著名的，高性能的算法。这些列于表4。

SATenstein-LS可以实例化模拟所有11的这些挑战者算法，所不同的是它不支持所使用的Ranov，G2和AG20预处理部件。所有我们的实验比较是基于原始算法的实现中，提交给相应的SAT比赛，除了PAWS，的量，UBCSAT执行几乎与在运行方面的原始解算器。所有比较是基于与每运行600秒的截止时间运行的每个解算器的10倍。

我们进行了我们的实验中55双3.2GHz的英特尔至强电脑具备2MB缓存和2GB RAM，运行的OpenSuSE 10.1的Linux集群上;运行时间为所有的算法（包括FocusedILS）进行测量CPU时间对这些参考的机器。



**5 结论**

在我们的每一个基准分布，我们能够发现，跑赢全部11个挑战者一个SATenstein-LS配置。我们的结果总结在表5中。

在惩罚平均运行时（即，我们明确地优化ParamILS目标函数，尽管根据我们使用我们的试验运行600秒截止，而不是我们在训练中使用的5-第二截止计算）而言，SATenstein-LS实现比挑战者明显更好的性能。对于QCP，HGEN和CBMC（S​​E），SATenstein-LS实现要比各自的最佳挑战者更好的PAR订单。对于SW-GCP，R3SAT和FAC，SATenstein-LS的性能优势仍然显着，但不戏剧性。这是不是太奇怪，有相对小的改进空间上R3SAT：这种类型的随机实例都突出使用在过去的17年评估SLS求解器的SAT，无论是在发展和竞争。相反，CBMC（S​​E）是一大类实例上SLS算法一般被认为表现疲软相比，公国家的最先进的全数字锁相环为基础的SAT求解器的新标杆。我们惊讶地看到改进，我们可以实现对QCP，相对广泛使用的基准测试，并HGEN，硬随机分布情况颇为相似R3SAT量。在这两种情况下，我们的研究结果表明，最近的事态发展SLS求解器的SAT尚未在旧的求解器（ANOV和SAPS / RSAPS，分别）产生显著的改善;因此，我们在这里属性SATenstein-LS的强劲表现给SLS求解器的发展在过去七年一直没有采取这些类型的基准考虑，也没有得到全面的改善董事会的事实。

我们还根据相比PAR等措施调查SATenstein-LS的表现。中位数的中值运行时（横跨整个10次运行在一个给定的实例中值的情况下，中位数）提供高性能的画面无视，只要大多数情况下都解决了大多数运行封顶运行。虽然SATenstein-LS没有被配置，以优化此性能测量，用它作为评价的基础产生的结果基本上类似的那些方法的PAR（与R3SAT是唯一的基准上几个挑战者打进稍好）。最后，我们测量实例在其上运行时中值是用于封盖运行截止下面的百分比。根据这个标准，我们SATenstein-LS求解是成功的在我们的每一个基准集实例的100％;另一方面，只有三个11挑战者在每个基准组解决了实例的40％以上。

挑战者的相对性能在不同的分布显著改变。例如，三个动态局部搜索算法（SAPS，RSAPS和爪子）执行都基本上比任何对保问题的其他挑战者（FAC）的更好;然而，在小世界图着色问题（SW-GCP）的相对性能较弱。同样，GNOV（随机SAT类别2007年SAT比赛的冠军）的表现非常糟糕我们的软件验证（CBMC（S​​E））和保理（FAC）的基准，但很有效地解决了SW-GCP和HGEN实例。 （有趣的是，在这两种类型的随机的SAT IN-立场我们考虑，GNOV没有达到的一些较老的SLS求解器的性能，特别是，爪子和RSAPS。）这表明，不太令人惊讶-，不同类型的SAT的基准最有效的解决，而采用不同的解决者。我们SATenstein-LS求解器进行比任何挑战者为我们的每一个分布的更好的事实清楚地表明，设计空间大集的高性能算法的特点跨越含有比以前了解得解算器，以及自动探索这片广袤的组合空间，可以有效地找到这样的改进设计。当然，对于个别基准制定解决者原则上可以使用基于实例的算法选择技NIQUE（如SATzilla）结合，产生更强大的性能。

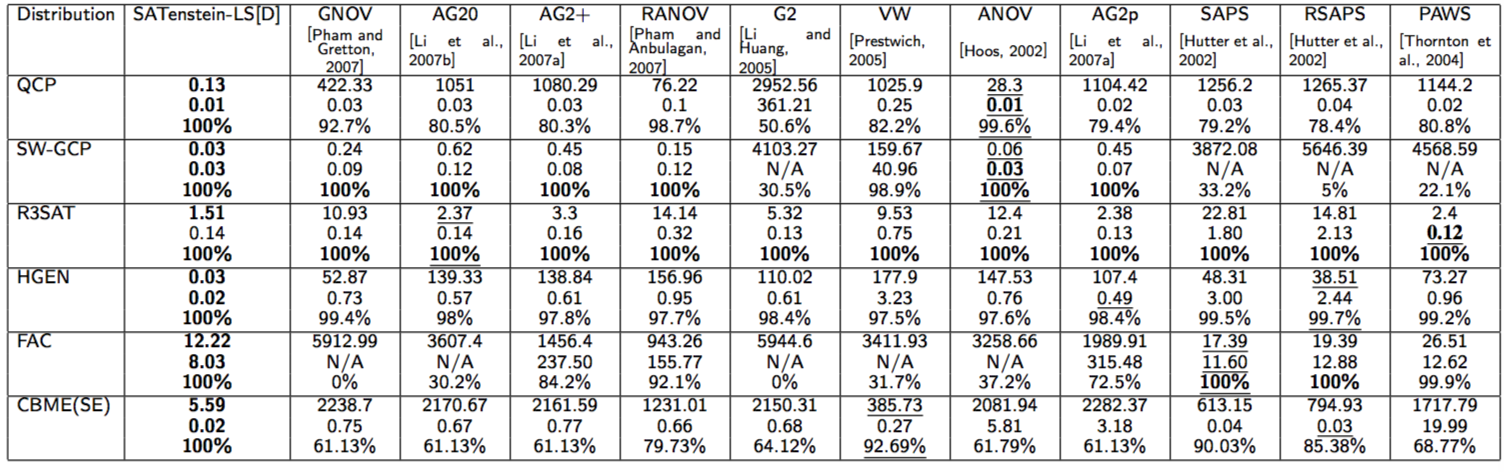


表5：SATenstein-LS的性能摘要和11挑战者。每个算法运行10次的每个实例，每运行600 CPU秒截止。每个单元⟨i，j⟩总结算法的J分布上我作为A / B / C，其中一个（上图）的测试集的表现是在处罚的平均运行时间; B（中）是位运行时间超过所有实例（没有定义如果少于一半的中间奔跑未能终止）的中位数; C（下）是解决实例百分比（即其平均运行时间<截止）。最好的计分算法每行中（多个）以粗体表示，并另外最好得分挑战者错误用下划线。

我们已经讨论了性能指标，迄今为止只描述总体性能在整个测试集。人们可能不知道是否我们SATenstein-LS求解的许多测试实例表现不佳，但对于这个补偿

虚弱的其他实例。表6示出，这是典型的情况并非如此，在每个实例的基础每个SATenstein-LS解算器的性能进行比较，以每挑战者。该SATenstein-LS求解跑赢上所有的基准集，除了R3SAT绝大多数测试实例的最佳挑战者。爪子是挑lenger即优于我们SATenstein-LS求解最频繁（的时间62％）。图1显示了这两种算法的性能是相当高度相关，并且，虽然是相当容易的两种算法实例倾向于更快爪子，SATenstein-LS [R3SAT]进行更好上更难实例解决。我们观察到在R3SAT其他挑战者同质化的趋势。这种现象更为明显为QCP和SW-GCP，但似乎没有发生为CBMC（S​​E）HGEN和FAC，其中在求解器性能的相关性也相当弱。

我们的惩罚平均运行措施是测试截止时间，这台点球选择敏感。特别是，算法可以取得好成绩时，截止时间是很大，但能达到弱得多PAR分值较小的临界值。令人欣慰的是，我们发现，并没有出现在我们的研究中考虑了SATenstein-LS求解这个问题。具体来说，SATenstein-LS求解器超过了其挑战者在PAR方面对我们所有的分布无论取舍门槛的使用时间，并从表5中同样的定性结果不受截止时间。图2给出了我们检查得出这些结论，考虑实例分布FAC的数据的一个例子。我们看到，而截止时间的选择影响了原料PAR分数，因为没有截止将一个挑战者都打出SATenstein-LS求解。 PAR先升，然后用截止下降，因为。增加的截止提高了罚款未解决的情况下（也因此而PAR），但会降低分数解决的情况下（因为处罚之前封顶的运行被替换的真正运行时） 。一旦给定的测试集中的所有实例都解决了，PAR保持恒定截止增加。

**6后续**

在这项工作中，我们主张，基于（1）一个框架，它能够灵活地组合来自现有高性能求解部件，和（2）的通用算法配置工具查找实例construct- ING启发式算法一个新的方法，对实例的给定集合表现良好。我们把这种方法用于随机局部搜索算法为SAT和经验证明，它能够产生代表在解决SAT实例的几个被广泛研究的分布相当大的改善优于现有技术的前一个状态的新SLS为基础的解决者。源代码，并为我们的SATenstein-LS框架的文档都可以在网上http://www.cs.ubc.ca/实验室/测试/项目/ SATenstein。

不同于弗兰肯斯坦博士从玛丽·雪莱的小说，他的怪异生物困扰着他够解渴永远他的野心，以创造一个“完美”人类的悲剧人物，我们感到鼓舞，释放不仅是我们新的求解器，而且还的全部功能我们自动解算器建设进程到其他类SAT基准。就像弗兰肯斯坦博士，我们发现我们的创作有点可怕，认识到我们的SATenstein求解器并不总是体现最元素GANT设计。因此，我们目前正在朝着我们SATenstein解算器如何与先前已知的SAT算法更详细的了解。未来工作其他有趣的线包括我们的解算器框架的延伸来捕获从G2WSAT架构和动态局部搜索算法的组件，以及预处理器（如使用的，例如在Ranov）的组合;通过算法选择的方法手段培训了各类SAT实例SATenstein求解器的组合;和比我们的方法的情况下ParamILS其他算法配置过程的调查。最后，通过实现对SLS-算法的SAT的结果的鼓舞，我们认为后面SATenstein一般的方法同样适用于非SLS基于解算器和其他的组合问题。套用玛丽·雪莱的维克多·弗兰肯斯坦的话，我们希望最终很多有效的解决者将欠他们的存在这个老本行。

参考文献

[Carchrae and Beck, 2005] T. Carchrae and J.C. Beck. Applying machine learning to low knowledge control of optimization algo-

rithms. *Computational Intelligence*, 21(4):373–387, 2005.

[Chiarandini *et al.*, 2008] M. Chiarandini, C. Fawcett, and H.H. Hoos. A modular multiphase heuristic solver for post enroll- ment course timetabling (extended abstract). In *Proc. PATAT*, 2008.

[Clarke *et al.*, 2004] E. Clarke, D. Kroening, and F. Lerda. A tool for checking ANSI-C programs. In *Proc. TACAS*, pages 168–176, 2004.

[Ee ́nandBiere,2005] N.Ee ́nandA.Biere.Effectivepreprocessing in SAT through variable and clause elimination. In *Proc. SAT*, pages 61–75, 2005.

[Fukunaga, 2002] A.S. Fukunaga. Automated discovery of com- posite SAT variable-selection heuristics. In *Proc. AAAI*, pages 641–648, 2002.

[Fukunaga, 2004] A.S. Fukunaga. Evolving local search heuristics for SAT using genetic programming. In *Proc. GECCO*, pages 483–494, 2004.

[GaglioloandSchmidhuber,2006] M.GaglioloandJ.Schmidhuber. Learning dynamic algorithm portfolios. *Annals of Mathematics and Artificial Intelligence*, 47(3-4):295–328, 2006.

[Gaspero and Schaerf, 2007] L.D. Gaspero and A. Schaerf. Easysyn++: A tool for automatic synthesis of stochastic local search algorithms. In *Proc. SLS*, pages 177–181, 2007.

[Gent*etal.*,1999] I.P.Gent,H.H.Hoos,P.Prosser,andT.Walsh. Morphing: Combining structure and randomness. In *Proc. AAAI*, pages 654–660, 1999.

[Gomes and Selman, 1997] C. P. Gomes and B. Selman. Problem structure in the presence of perturbations. In *Proc. AAAI*, pages 221–226, 1997.

[Gomes and Selman, 2001] C. P. Gomes and B. Selman. Algorithm portfolios. *Artificial Intelligence*, 126(1-2):43–62, 2001.

[Gratch and Dejong, 1992] Jonathan Gratch and Gerald Dejong. COMPOSER: A probabilistic solution to the utility problem in speed-up learning. In *Proc. AAAI*, pages 235–240, 1992.

[GuerriandMilano,2004] A.GuerriandM.Milano.Learningtech- niques for automatic algorithm portfolio selection. In *Proc. ECAI*, pages 475–479, 2004.

[Hirsch, 2002] E. A. Hirsch. Random generator hgen2 of satisfi- able formulas in 3-CNF. http://logic.pdmi.ras.ru/

̃hirsch/benchmarks/hgen2-1.01.t%ar.gz, 2002. [Hoos, 2002] H.H. Hoos. An adaptive noise mechanism for Walk-

SAT. In *Proc. AAAI*, pages 655–660, 2002.

[Hoos, 2008] Holger H. Hoos. Computer-aided design of high- performance algorithms. Technical Report TR-2008-16, Uni- versity of British Columbia, Department of Computer Science, 2008.

[Hutter*etal.*,2002] F.Hutter,D.A.D.Tompkins,andH.H.Hoos. Scaling and probabilistic smoothing: Efficient dynamic local search for SAT. In *Proc. CP*, pages 233–248, 2002.

[Hutter *et al.*, 2007a] F. Hutter, D. Babic ́, H.H. Hoos, and A.J. Hu. Boosting verification by automatic tuning of decision procedures. In *Proc. FMCAD*, pages 27–34, 2007.

[Hutter *et al.*, 2007b] F. Hutter, H. H. Hoos, and T. Stu ̈tzle. Au- tomatic algorithm configuration based on local search. In *Proc. AAAI*, pages 1152–1157, 2007.

[Hutter*etal.*,2008] F.Hutter,H.H.Hoos,T.Stu ̈tzle,andK.Leyton- Brown. ParamILS version 2.2. http://www.cs.ubc.ca/ labs/beta/Projects/ParamILS, 2008.

[KhudaBukhsh, 2009] A.R. KhudaBukhsh. SATenstein: Auto- matically building local search SAT solvers from components. http://www.cs.ubc.ca/labs/beta/Projects/ SATenstein/ashiq%ue\_masters\_thesis.pdf, 2009.

[Li and Huang, 2005] C.M. Li and W. Huang. Diversification and determinism in local search for satisfiability. In *Proc. SAT*, pages 158–172, 2005.

[Li *et al.*, 2007a] C. M. Li, W. X. Wei, and H. Zhang. Combin- ing adaptive noise and look-ahead in local search for SAT. In *Proc. SAT*, pages 121–133, 2007.

[Li *et al.*, 2007b] C.M. Li, W. Wei, and H. Zhang. Combining adap- tive noise and promising decreasing variables in local search for SAT. Solver description, SAT competition 2007, 2007.

[McAllester *et al.*, 1997] D. McAllester, B. Selman, and H. Kautz. Evidence for invariants in local search. In *Proc. AAAI*, pages 321–326, 1997.

[Minton, 1993] S. Minton. An analytic learning system for special- izing heuristics. In *Proc. IJCAI*, pages 922–929, 1993.

[Monette*etal.*,2009] J.Monette,Y.Deville,andP.V.Hentenryck. Aeon: Synthesizing scheduling algorithms from high-level models. In *Proc. INFORMS (to appear)*, 2009.

[Oltean, 2005] M. Oltean. Evolving evolutionary algorithms us- ing linear genetic programming. *Evolutionary Computation*, 13(3):387–410, 2005.

[Pham and Anbulagan, 2007] D.N. Pham and Anbulagan. Resolu- tion enhanced SLS solver: R+AdaptNovelty+. Solver description, SAT competition 2007, 2007.

[Pham and Gretton, 2007] D. N. Pham and C. Gretton. gNovelty+. Solver description, SAT competition 2007, 2007.

[Prestwich, 2005] S. Prestwich. Random walk with continuously smoothed variable weights. In *Proc. SAT*, pages 203–215, 2005.

[SAT Competition, 2009] SAT Competition. http://www. satcompetition.org, 2009.

[Selman *et al.*, 1994] B. Selman, H. Kautz, , and B. Cohen. Noise strategies for improving local search. In *Proc. AAAI*, pages 337– 343, 1994.

[Simon, 2002] L. Simon. SAT competition random 3CNF generator. http://www.satcompetition.org/2003/ TOOLBOX/genAlea.c, 2002.

[Thornton*etal.*,2004] J.Thornton,D.N.Pham,S.Bain,andV.Fer- reira. Additive versus multiplicative clause weighting for SAT. In *Proc. AAAI*, pages 191–196, 2004.

[TompkinsandHoos,2004] D.A.D. Tompkins and H.H. Hoos. UBCSAT: An implementation and experimentation environment for SLS algorithms for SAT & MAX-SAT. In *Proc. SAT*, pages 37–46, 2004.

[Uchida and Watanabe, 1999] T. Uchida and O. Watanabe. Hard SAT instance generation based on the factorization prob- lem. http://www.is.titech.ac.jp/ ̃watanabe/ gensat/a2/GenAll.ta%r.gz, 1999.

[Westfold and Smith, 2001] S.J. Westfold and D.R. Smith. Synthe- sis of efficient constraint-satisfaction programs. *The Knowledge Engineering Review*, 16(1), 2001.

[Xu*etal.*,2008] L.Xu,F.Hutter,H.H.Hoos,andK.Leyton-Brown. SATzilla: portfolio-based algorithm selection for SAT. *Journal of Artificial Intelligence Research*, 32:565–606, 2008.

[Xu*etal.*,2009] L.Xu,F.Hutter,H.H.Hoos,andK.Leyton-Brown. SATzilla2009: An automatic algorithm portfolio for SAT. Solver description, 2009 SAT Competition, 2009.