2018 IEEE 30th International Conference on Tools with Artificial Intelligence

Constrainedness in Stable Matching

Guillaume Escamocher
Insight Centre for Data Analytics
University College Cork, Ireland
guillaume.escamocher@insight-centre.org

Barry O'Sullivan
Insight Centre for Data Analytics
University College Cork, Ireland
barry.osullivan@insight-centre.org

***Abstract*—In constraint satisfaction problems, constrainedness provides a way to predict the number of solutions: for instances of a same size, the number of constraints is inversely correlated with the number of solutions. However, there is no obvious equivalent metric for stable matching problems. We introduce the *contrarian score*, a simple metric that is to matching problems what constrainedness is to constraint satisfaction problems. In addition to comparing the contrarian score against other potential tightness metrics, we test it for different instance sizes as well as extremely distinct versions of the stable matching problem. In all cases, we find that the correlation between contrarian score and number of solutions is very strong.**

I. Introduction

Constrainedness, or tightness, is the density of constraints in a constraint satisfaction problem (CSP) instance. It is a simple metric, computable in linear time with respect to the size of an instance, yet can be used to reveal a lot of information about the problem. Several metrics based on constrainedness have been advanced to characterize the complexity of CSP instances. Some rely on tightness directly [9], [10], while others use an intermediate tool like backdoors [19] to provide insight into instance hardness [7], [15].

Constrainedness is directly related to the number of solutions. Adding a constraint to a CSP instance cannot increase its number of solutions, while removing a constraint cannot decrease that number. Therefore, constrainedness acts as a predictor of the number of solutions for constraint problems: a higher tightness means a lower expected number of solutions.

Stable matching is the problem of grouping some agents according to their preferences, such that the agents have no incentive to change the grouping. It has a plethora of applications, the most frequently cited are the assignment of students to universities, and of residents to hospitals [14].

Two-Dimensional Stable Matching (2DSM), commonly referred to as "stable marriage", is the matching problem where agents belong to one of two sets and have preferences over the agents from the other set. A solution is a set of pairs where each pair contains one agent from each agent set, each agent belongs to exactly one pair, and no two agents from distinct pairs would rather be together than stay with their assigned partner. 2DSM was introduced in 1962 as the first example of stable matching problem [8] and is still one of the most studied matching problems, if not the most. Perhaps explaining its popularity, it is an easy problem: every 2DSM instance has at least one solution. Furthermore, 2DSM instances can be solved in time quadratic in the size of the agent sets [8].

In this paper we introduce a tightness-like tool for the stable matching problem. Our metric, the *contrarian score*, is easy to compute and, as we will show, also correlates with the number of solutions. To choose our tightness metric for stable matching, we looked at the properties held by matching instances with an extreme number of stable matchings, either low or high. Matching instances with master preference lists are often the ones with the fewest amount of solutions, whether in 2DSM [13] or in some other stable matching problems [6]. For these instances all agents within the same agent set share the same preference list.

On the other end of the spectrum, 2DSM instances with a high number of solutions (exponential in the size of the agent sets) have been found using Latin squares as preference matrices [1]. A key constraint in the construction of these instances requires each agent to rank every agent from the other agent set the opposite way that they are ranked by that other agent. So if for example agent a ranks agent b on second position of its preference list, then b must rank a second-to-last of its own preference list. This balances the preferences around, to the contrary of master preference lists.

Both kinds of instances seem to indicate the same trend: more balanced preferences yield more solutions. We present three different interpretations of the notion of balance, and associate a quantification metric to each one. First is the split score, which counts how many times a given agent is preferred over the other agents from its own agent set. Next is the Latin score, which measures instances based on how close their preference matrices are to Latin squares. Last is the contrarian score, which increases when the rankings that two given agents give each other are opposite. As we show in the paper, the contrarian score is the one that has by far the strongest correlation with the number of solutions.

After establishing that the contrarian score is a valid measure of constrainedness for 2DSM instances, we examine its behavior on matching problems with three agent sets. Three-sided matching can be used in a number of practical applications, examples of which include computer networking [3], market strategy [17] and kidney exchange [2]. There are multiple variants of three-sided matching problems; we look at two of them in particular: Three-Dimensional Stable Matching (3DSM) and 3DSM with strong stability (3DSM$_{strong}$).

This publication has emanated from research conducted with the financial support of Science Foundation Ireland (SFI) under Grant Number SFI/12/RC/2289, co-funded under the European Regional Development Fund.

2375-0197/18/$31.00 ©2018 IEEE
DOI 10.1109/ICTAI.2018.00112

3DSM is an extension of 2DSM to three agent sets, with an equivalent definition of stability. The question of whether every 3DSM instance admits a solution is still open, however it is strongly conjectured that the answer is positive [5]. The interest of testing our metric on this problem resides in the fact that 3DSM instances have many more solutions than 2DSM instances [6]. For example, Figures 8 and 9 will illustrate that 3DSM instances with 7 agents in each agent set have about a thousand times as many solutions as their 2DSM counterparts with 10 agents in each set.

3DSM$_{strong}$ instances are identical to 3DSM ones, however the definition of stability is much stronger, greatly decreasing the number of solutions. The interest of this problem resides instead in its difficulty. 3DSM$_{strong}$ instances are harder to solve than instances from the other problems that we considered. While all 2DSM instances have at least one solution, and while 3DSM instances seem to have a large number of solutions, not all 3DSM$_{strong}$ instances even have a solution. In fact, the problem of determining whether a given 3DSM$_{strong}$ instance admits a solution is NP-Complete [11].

II. General Definitions

We begin by defining the exact kind of matching instances that we consider in this paper.

Definition 1: A *two-dimensional stable matching instance*, or *2DSM instance*, comprises:

- Two *agent sets* $A = \{a_1, a_2, \ldots, a_n\}$ and $B = \{b_1, b_2, \ldots, b_n\}$, each containing n agents.
- For each agent $a \in A$, a strict preference order $>_a$ over the agents from the set B. For each agent $b \in B$, a strict preference order $>_b$ over the agents from the set A.

2DSM is often referred to as "stable marriage". We will also look at a three-sided extension of stable marriage, which is defined in a similar way:

Definition 2: A *three-dimensional stable matching instance*, or *3DSM instance*, comprises:

- Three agent sets $A = \{a_1, a_2, \ldots, a_n\}$, $B = \{b_1, b_2, \ldots, b_n\}$ and $C = \{c_1, c_2, \ldots, c_n\}$, each containing n agents.
- For each agent $a \in A$, a strict preference order $>_a$ over the agents from the set B. For each agent $b \in B$, a strict preference order $>_b$ over the agents from the set C. For each agent $c \in C$, a strict preference order $>_c$ over the agents from the set A.

The number of agents in each agent set is the *size* of the instance. We show a size 4 instance of each problem in Figures 1 and 2 respectively. In the figures, every agent is followed by its ordered preference list, with the favorite coming first. These instances will be used multiple times troughout the paper to illustrate other definitions.

An instance can be viewed as the expression of agent preferences. In this paper "dDSM instance" will refer to a d-dimensional stable matching instance where $d = 2$ or $d = 3$. Assigning each agent from A to exactly one agent from B and (if $d = 3$) exactly one agent from C is matching them:

a_1:	$\boxed{b_3}$	b_2	b_4	b_1		b_1:	a_1	$\boxed{a_4}$	a_2	a_3
a_2:	b_4	$\boxed{b_2}$	b_3	b_1		b_2:	a_3	a_4	a_1	$\boxed{a_2}$
a_3:	$\boxed{b_4}$	b_2	b_3	b_1		b_3:	a_3	a_4	$\boxed{a_1}$	a_2
a_4:	b_3	b_4	$\boxed{b_1}$	b_2		b_4:	$\boxed{a_3}$	a_4	a_2	a_1

Fig. 1. A 2DSM instance with a split score of 11, a Latin score of 17 and a contrarian score of 84.

a_1:	b_3	b_4	b_2	$\boxed{b_1}$		b_1:	c_4	c_2	c_3	$\boxed{c_1}$
a_2:	b_4	$\boxed{b_3}$	b_1	b_2		b_2:	c_2	$\boxed{c_4}$	c_1	c_3
a_3:	$\boxed{b_2}$	b_1	b_3	b_4		b_3:	c_1	c_3	$\boxed{c_2}$	c_4
a_4:	b_1	b_2	$\boxed{b_4}$	b_3		b_4:	$\boxed{c_3}$	c_1	c_4	c_2

c_1:	a_4	a_3	a_2	$\boxed{a_1}$
c_2:	$\boxed{a_2}$	a_1	a_4	a_3
c_3:	a_3	$\boxed{a_4}$	a_1	a_2
c_4:	a_1	a_2	$\boxed{a_3}$	a_4

Fig. 2. A contrarian 3DSM instance of size 4.

Definition 3: Let I be a dDSM instance of size n. A *matching for* I is a set M of n d-tuples such that each d-tuple of M contains one element from each of the d agent sets of I, and each agent of I occurs exactly once in M.

Figures 1 and 2 present a matching example for each d. Each agent there is assigned to the agent from its preference list indicated by a square.

A matching achieves stability when no d agents wish to deviate from the matching and form their own tuple together:

Definition 4: Let I be a dDSM instance, let M be a matching for I and let t be a d-tuple not in M such that t contains an agent from each agent set of I. We say t is a *blocking tuple for* M if one of the following conditions is true:

- $d = 2$, $t = \langle a_i, b_j \rangle$, a_i ranks b_j ahead of the agent from B it got assigned to in M, and b_j ranks a_i ahead of the agent from A it got assigned to in M.
- $d = 3$, $t = \langle a_i, b_j, c_k \rangle$, a_i ranks b_j ahead of the agent from B it got assigned to in M, b_j ranks c_k ahead of the agent from C it got assigned to in M and c_k ranks a_i ahead of the agent from A it got assigned to in M.

Definition 5: Let I be a dDSM instance and let M be a matching for I. We say that I is a *solution* for I if there is no blocking tuple for M.

The matching from Figure 1 is not a solution because $\langle a_4, b_3 \rangle$ is a blocking pair. Indeed, a_4 prefers b_3 over its matching partner b_1 and b_3 prefers a_4 over its matching partner a_1. On the other hand there is no blocking triple for the matching from Figure 2, therefore that matching is a solution for the instance depicted.

Solutions are also called "stable matchings" in the literature. 3DSM instances have in general many more solutions than 2DSM ones. The 3DSM instance from Figure 2 has 194 solutions, while the most solutions for a 2DSM instance of the same size is 10 [4].

Stability in the Matching Problem can take several forms. Using an alternative definition of stability, one can find a completely different set of solutions for the exact same instance. To

ensure that our proposed contrarian metric performs well on a wide variety of problem variations, we will make experiments on another version of stability, which is based on a relaxed definition of blocking tuple for 3DSM:

Definition 6: Let I be a 3DSM instance, let M be a matching for I and let $t = \langle a_i, b_j, c_k \rangle$ be a triple not in M. Let b_M, c_M and a_M the agents respectively assigned to a_i, b_j and c_k in M. We say that t is a *weakly blocking tuple for* M if one of the following conditions is true:

- $b_j >_{a_i} b_M$, $c_k \geq_{b_j} c_M$ and $a_i \geq_{c_k} a_M$.
- $b_j \geq_{a_i} b_M$, $c_k >_{b_j} c_M$ and $a_i \geq_{c_k} a_M$.
- $b_j \geq_{a_i} b_M$, $c_k \geq_{b_j} c_M$ and $a_i >_{c_k} a_M$.

In other words, a triple t is weakly blocking if one of its agents prefers t over its matching triple, and the other two are at least indifferent. We call 3DSM$_{strong}$ the three-sided matching problem which uses this stronger form of stability. Works on 3DSM$_{strong}$ sometimes call *strongly blocking tuple* the object from Definition 4 and *weak stability* the resulting notion of stability. In a similar manner as for weak stability, a *solution* for a 3DSM$_{strong}$ instance is a matching that does not have any weakly blocking triple.

Note that while every solution to a 3DSM$_{strong}$ instance will also be a solution to that same instance under weak stability, the reverse is not generally true. Only 33 of the 194 solutions to the 3DSM instance from Figure 2 remain solutions for the same instance in the 3DSM$_{strong}$ problem. In particular, the matching from Figure 2 is a solution under weak stability but not under strong stability. Indeed, consider the triple $t = \langle a_1, b_3, c_1 \rangle$. Both a_1 and b_3 would rather be in t than with their respective matching partners, however c_1 is assigned to a_1 either way. This makes t weakly blocking for the matching shown, but not strongly.

III. A Tightness Metric for Stable Matching

A. Methodology of the Experiments

For each experiment, we generate instances with a low metric score, and then switch agents in the preference lists to increase the score until we reach a local maximum.

For the initial instances, we generate instances with shared (master) preference lists and switch two random agents in one preference list in each of the d agent sets. Instances with master preference lists often are the instances with the lowest possible metric score, or close to it [6]. However, all such instances of a given size are isomorphic, so we add an initial random switch in each agent set to get random starting points.

At each step, there are $dn^2(n-1)/2$ possible switches: $n(n-1)/2$ switches for each preference list, and dn preference lists. We look at all of them and keep the one that increases the metric score the most. If no single switch strictly increases the metric score, we look at all possible pairs of two switches. If several switches (or pairs of switches) increase the metric score by the same amount, we pick a random one. When no switch or pair of switches can increase the metric score any further, we stop the run. Each experiment consists of 100 runs.

We present the results of each experiment through two plots. In the first one each point represents a step s, with the average number of solutions for the instances obtained at step s plotted against their average metric score. The number of solutions is computed using Cachet, an exact SAT model counter [16].

Depending on the initial instances, and which switches are subsequently made, some runs take longer than others to reach a local maximum. The first plot only contains steps that are reached by at least a quarter of the runs (25).

The second plot compares the number of solutions against the relevant metric score for every individual instance measured. This includes those from the end of very long runs that were removed from the first plot, as well as 100 completely random instances that are generated as a control experiment.

We want a metric which score describes the relaxation of the constraints. So matching instances with a higher score are expected to have more solutions in average, in the same way that constraint instances with more relaxed constraints generally have more solutions. Therefore the desired behavior of the plots is to show a gradual increase, as monotonic as possible, from instances with a low score and few solutions to instances with a high score and many solutions.

B. Finding the Best Metric

We now empirically test different definitions of metric, in order to find the one that approximates constrainedness the closest. In a previous stable matching paper [6] we introduced the notion of *perfectly split* instance. For each pair of agents belonging to the same agent set of a perfectly split instance, each agent of the pair is favored over the other by exactly half (give or take one for odd sizes) of the agents that rank them. The instances most dissimilar to perfectly split instances are the ones with master preference lists. In these instances, preferences are never split and always absolute.

The main experiment from that paper started from instances with master preference lists and at each step randomly picked two agents that were not equally favored, and switched them in a preference list to decrease the difference. We found out that the initial instances with master preference lists, the ones that are the furthest away from perfectly split instances, are among the most constrained matching instances, the ones with the fewest solutions. Unfortunately instances at the other end, the ones that are perfectly split or close to it, turned out to be indistinguishable from random instances in terms of number of solutions. To see if the idea can still be salvaged, we refine the notion with the split score, a quantifier of the distance to a perfectly split instance.

Definition 7: Let I be a dDSM instance of size n. Let p and p' be two agents from the same agent set in I where p is preferred over p' as least as often as p' is preferred over p. The *split score* of pair (p, p') is the number of times p' is preferred over p. The *split score* of I is the sum of split scores for all $dn(n-1)/2$ pairs of agents within a same agent set.

The split score of a dDSM instance of size n can range from 0 (in every pair of agents from the same agent set, one of the agents is always favored over the other) to $dn(n-1)/2 \times \lfloor n/2 \rfloor$ (the preferences between any two agents from the same agent set are as evenly split as possible).

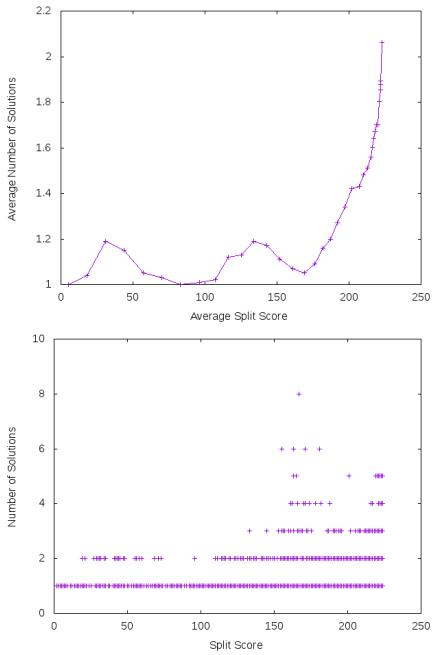

Fig. 3. Correlation between split score and number of solutions for 2DSM instances of size 8. Averages (top) and scatter plot (bottom).

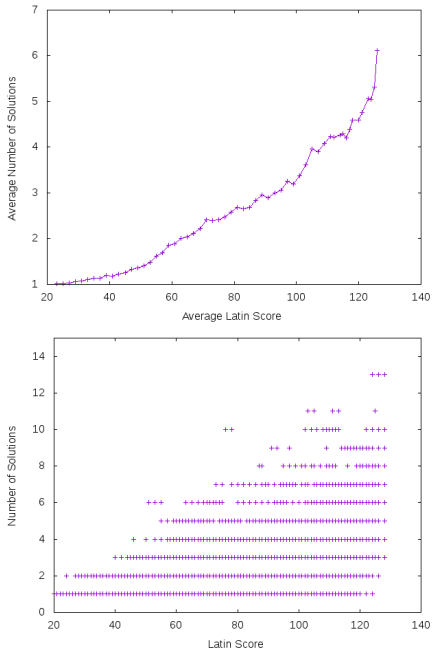

Fig. 4. Correlation between Latin score and number of solutions for 2DSM instances of size 8. Averages (top) and scatter plot (bottom).

We illustrate the notion on the 2DSM instance from Figure 1. The split score for the pair (b_1, b_2) is 1 because the least preferred agent of the two, b_1, if favored once over b_2. Similarly, the split score for the pair (a_2, a_4) is 0 because a_2 is never favored over a_4 and the split score of the pair (b_2, b_3) is 2 because the preferences between these two agents are evenly split. The sum of the twelve pairs' split scores is 11, this is the split score of the instance.

We present the results of our split score experiments in Figure 3. As the figure shows, the split score is not a valid measure of constrainedness for matching instances. Not only is the correlation inexistent, but most instances generated have a very low number of solutions (2 or less), even as the metric score grows. We therefore need to look in another direction to quantify tightness in stable matching.

We now consider the Latin score, which measures how close the preference matrices are to Latin squares.

Definition 8: Let I be a dDSM instance. We note $Col_{A,i}$ (respectively $Col_{B,i}$ and $Col_{C,i}$ if $d = 3$) the set of agents that are the i^{th} choice of at least one agent from A (respectively B, C). These dn sets are the *columns* of I. The *Latin score* of a column is the number of agents that it contains. The *Latin score* of I is the sum of the dn column Latin scores.

The Latin score of a dDSM instance of size n can range from dn (all agents in the same agent set have the same preference list) to dn^2 (the preference matrices are Latin squares). As an example, we look again at the instance from Figure 1. In this case, the column Latin scores range from 1 (for $Col_{B,2} = \{a_4\}$) to 3 (for $Col_{A,3} = \{b_4, b_3, b_1\}$ and $Col_{B,4} = \{a_3, a_2, a_1\}$). The eight column Latin scores sum to 17, which is the Latin score of the instance.

Instances with master prefrence lists have the lowest possible Latin score (dn). On the other hand, efforts to generate matching instances with many solutions have focused on Latin and pseudo-Latin constructions of the preference matrices. Therefore, the Latin score seems like a good candidate to represent constrainedness and predict the number of solutions for matching instances. We present the results of our Latin score experiments in Figure 4.

The part of the figure plotting the averages seems to indicate a correlation, albeit quite weak. Furthermore, the number of solutions grows significantly higher than when using the split score. However, the scatter plot reveals that there is a significant number of instances with a very high Latin score and very few solutions. If high scoring matching instances are to correspond to highly relaxed constraint instances, one would not expect to encounter this behavior. So while the Latin score is a clear improvement on the split score, it is not accurate enough in its assessment of constrainedness.

We now introduce the main metric of the paper, the contrarian score. We first define instances with a perfect contrarian score, simply called contrarian instances.

Definition 9: Let I be a dDSM instance of size n. We say that I is *contrarian* if for each agent p and each $1 \leq i \leq n$, one of the following conditions is true:

- $d = 2$ and the agent q whom p ranks i^{th} ranks p in $(n + 1 - i)^{th}$ position.
- $d = 3$ and the agent q who is ranked i^{th} by the favorite agent of p ranks p in $(n + 1 - i)^{th}$ position.

a_1:	b_2	b_3	b_1	b_4		b_1:	a_2	a_1	a_3	a_4
a_2:	b_3	b_2	b_4	b_1		b_2:	a_4	a_3	a_2	a_1
a_3:	b_4	b_1	b_2	b_3		b_3:	a_3	a_4	a_1	a_2
a_4:	b_1	b_4	b_3	b_2		b_4:	a_1	a_2	a_4	a_3

Fig. 5. A contrarian 2DSM instance of size 4.

Two contrarian instances are shown in Figures 5 (for 2DSM) and 2 (for 3DSM). As an example, the agent a_1 in the 2DSM instance ranks b_2 first so b_2 ranks it last, ranks b_3 second so b_3 ranks it second to last, etc. Similarly in the 3DSM example, the favorite of a_1 ranks c_1 first so c_1 ranks a_1 last, it ranks c_3 second so c_3 ranks a_1 second to last, etc.

Contrarian instances have led to the crafting of 2DSM instances with many solutions [1][1]. They exhibit a number of interesting properties. Their preference matrices are always Latin squares, and a whole contrarian instance can be retrieved from only one of its d preference matrices (modulo isomorphism when $d = 3$).

We now define the contrarian score, which is computed from the differences with regard to a fully contrarian instance.

Definition 10: Let I be a dDSM instance. The *contrarian score* of an agent p of I is $\sum_{i=1}^{n} n - c(p, i)$, where $c(p, i)$ is defined in the following way:

- when $d = 2$: let q be the agent ranked i^{th} by p and let j be the ranking of p by q. Then $c(p, i) = |n + 1 - i - j|$.
- when $d = 3$: let q be the agent ranked i^{th} by the favorite agent of p and let j be the ranking of p by q. Then $c(p, i) = |n + 1 - i - j|$.

The *contrarian score* of I is the sum of the contrarian scores of all dn agents of I.

A fully contrarian instance will have a contrarian score of dn^3. It is not clear what the lowest possible score is, however instances with master preference lists have a contrarian score of $d\sum_{i=1}^{n}(\sum_{j=1}^{n}(n-(|i-j|))) = 2d\binom{n+1}{3}$. As usual, Figure 1 illustrates the metric. Agent a_1 ranks b_3 first so b_3 should rank a_1 last, but b_3 ranks a_1 second to last instead, which is a difference of 1. Similarly, a_1 is correctly ranked third by b_2, is ranked fourth by b_4 instead of second and is correctly ranked first by b_1. So the contrarian score of a_1 is $4-1+4-0+4-2+4-0 = 13$. The preference scores of the other agents can be computed in the same way to arive to a total contrarian score of 84 for the whole instance.

Figure 6 shows the results of our contrarian score experiments on instances of the same problem and size than the split score and Latin score ones. Contrary to these metrics, the contrarian score performs extremely well. The correlation with the number of solutions is sharp, and the scatter plot confirms that all instances obtained neatly fall in the desired range. No single instance with very few solutions has a high contrarian score. Likewise, no single instance with many solutions has a low contrarian score. This shows that the contrarian score is an excellent predictor of the number of solutions and a strong candidate for representing constrainedness in matching, at least for stable marriage instances of this size.

[1]They are called *Latin marriages* in the source, but the restrictions imposed are not limited to Latin square preference matrices.

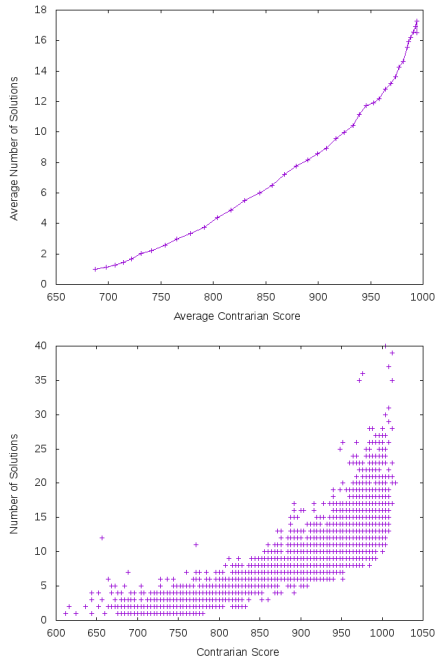

Fig. 6. Correlation between contrarian score and number of solutions for 2DSM instances of size 8. Averages (top) and scatter plot (bottom).

C. Different Sizes

Our contrarian score metric takes inspiration from the work that has been done on the generation of 2DSM instances with many solutions. Some methods that have been proposed in that field of study depend a lot on the size n of the instances considered. Early results were restricted to the case where n is a power of 2 [12]. Later constructions based on contrarian instances, the particular kind of instances our contrarian score metric takes inspiration from, were only defined for even instance sizes [1]. More recent approximations of $f_{\max}(n)$, the maximum possible number of solutions for 2DSM instances of size n, look at all instance sizes but still observe a behavior that differs according to the parity of n. As Table 3 in [18] indicates, $f_{\max}(n)$ seems to increase much less from $n = 2k$ to $n = 2k + 1$ than from $n = 2k + 1$ to $n = 2k + 2$.

For these reasons, it is important to verify that the behavior exhibited by the contrarian score can be replicated for other instance sizes. Our experiments thus far were on instances of size 8, which is not only even but a power of 2. We now repeat the same experiments for 2DSM instances of size 9 (represented in Figure 7) and 10 (represented in Figure 8).

For all three instance sizes we experimented on, the contrarian score yielded the same results: gradual increase in the number of solutions when increasing the metric. This shows the strong correlation between contrarian score and number of solutions is not dependent on a particular instance size.

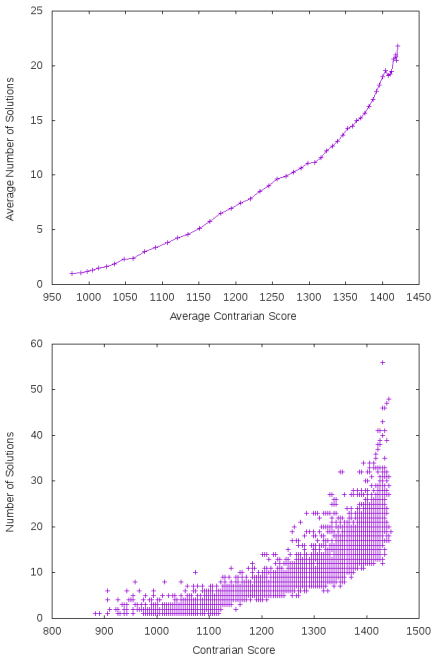

Fig. 7. Correlation between contrarian score and number of solutions for 2DSM instances of size 9. Averages (top) and scatter plot (bottom).

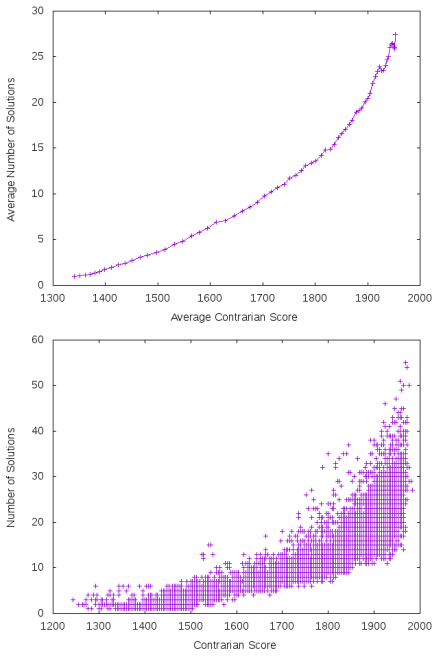

Fig. 8. Correlation between contrarian score and number of solutions for 2DSM instances of size 10. Averages (top) and scatter plot (bottom).

D. Different Problems

We have shown that the contrarian score performs well for the 2DSM problem. An interesting question is whether it can also be used for matching problems with radically different properties. We now test our metric on two additional matching problems that each differs from 2DSM in at least one significant way.

We start by experimenting on 3DSM instances. These instances have much more solutions than instances of comparable sizes from other matching problems. As before, we test different instance sizes. Our experiments are represented in Figure 9 for size 7 and in Figure 10 for size 8.

Surprisingly enough for a metric that was primarily designed for two-sided matching, the contrarian score performs as well for 3DSM as it does for 2DSM. The correlation plots are as sharp as they were in the previous experiments, and the scatter plots look even more concentrated in the desired region. This shows that the contrarian score can be used for another matching convention than 2DSM.

The other matching problem that we are studying is 3DSM$_{strong}$, which consists of the same instances as 3DSM but with a stricter definition of stability. Contrary to 2DSM instances, not all 3DSM$_{strong}$ instances admit a solution. This makes 3DSM$_{strong}$ a harder problem to solve.

We present our results in Figure 11 for size 8 and in Figure 12 for size 9. The correlation here seems a bit weaker than the one we observed for the other problems. The number of solutions does not increase substantially until instances reach a fairly high metric score. However, the general trend is

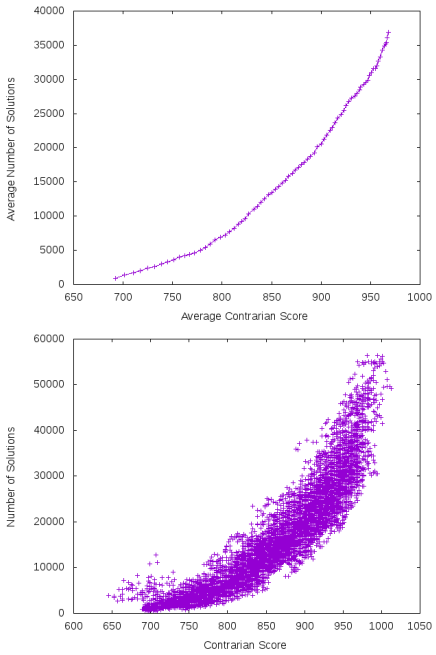

Fig. 9. Correlation between contrarian score and number of solutions for 3DSM instances of size 7. Averages (top) and scatter plot (bottom).

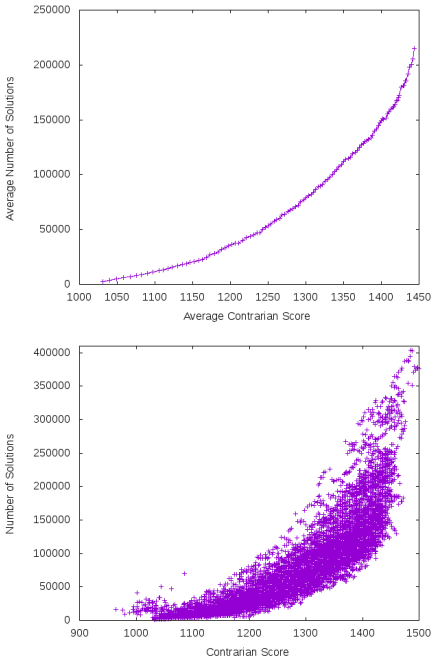

Fig. 10. Correlation between contrarian score and number of solutions for 3DSM instances of size 8. Averages (top) and scatter plot (bottom).

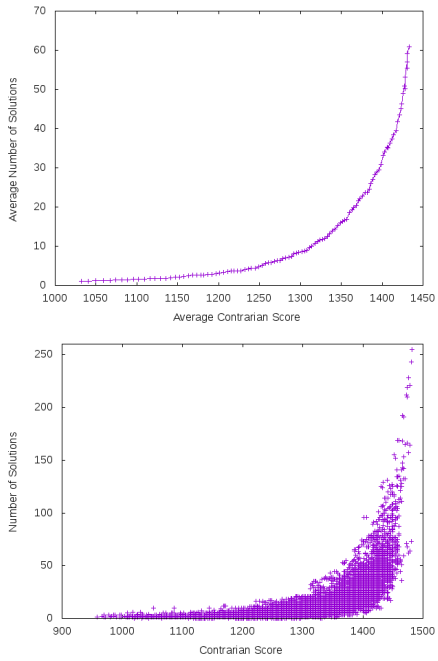

Fig. 11. Correlation between contrarian score and number of solutions for 3DSM$_{strong}$ instances of size 8. Averages (top) and scatter plot (bottom).

still present: the number of solutions monotonically increases with the contrarian score.

The experiments in this subsection show that while the contrarian score may not perfectly capture constrainedness in all matching variants, it can definitely be used in matching problems that considerably differ from standard 2DSM.

IV. Generating Instances with Many Solutions

In this section we explore a possible application of constrainedness for stable matching. We use local search heuristics based on a number of tightness metrics to find matching instances with as many solutions as possible.

We test four metrics. The first is the number of solutions. Since this metric is also the objective, we expect it to give the best results. The other three are the ones that we compared in the paper (split score, Latin score and contrarian score). Each experiment consists of 100 runs. Each run starts from a random instance and makes the switch that increases the most the relevant metric. All metrics share the same pool of 100 initial random instances. For each run, we only keep the instance with the most solutions. The experiments are very similar to the ones from the previous section, the main differences being the composition of the initial instances and that we are only interested in the instances with the highest scores.

We do not expect the contrarian score to seriously compete in this domain with the tailor-made 2DSM generation methods from [18], however there is a couple of reasons which make the experiments worthwhile. First, the work done on this topic focuses on 2DSM, and we know that the contrarian score is a valid measure for various matching problems. Our results on

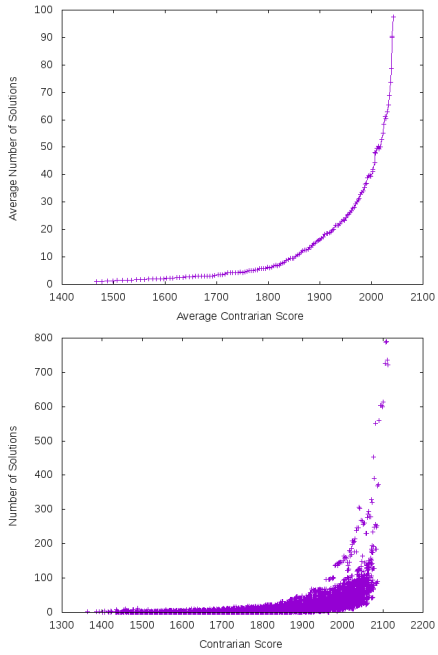

Fig. 12. Correlation between contrarian score and number of solutions for 3DSM$_{strong}$ instances of size 9. Averages (top) and scatter plot (bottom).

three-sided matching are in fact, to the best of our knowledge, the first of their kind for these particular problems.

Second, state of the art results rely on specific preference structures that can only provide a few instances for each size.

TABLE I
NUMBER OF SOLUTIONS OBTAINED USING DIFFERENT METRICS.

Metric	Run Rank	# Solutions		
		2DSM $n=7$	3DSM $n=6$	3DSM$_{strong}$ $n=7$
# Solutions	Best	30	8973	353
	Average	14	7743	89
	Worst	4	6614	11
Split score	Best	10	3287	18
	Average	3	1803	6
	Worst	1	927	1
Latin score	Best	14	3687	33
	Average	5	2878	15
	Worst	3	2156	4
Contrarian score	Best	22	8497	138
	Average	15	5341	50
	Worst	8	3855	16

TABLE II
COMPARISON OF RUNTIMES FOR DIFFERENT METRICS.

Metric	2DSM $n=7$	3DSM $n=6$	3DSM$_{strong}$ $n=7$
# Solutions	4h 11 mins	32d 2h 47 mins	1d 7h 23 mins
Split score	<1 min	45 mins	2 mins
Latin score	1 min	50 mins	5 mins
Contrarian score	3 mins	2h 15 mins	46 mins

These instances are artificial and often similar to each other. Breaking ties between equivalent switches at random, as we do, ensures on the other hand that the final instances we can obtain are numerous and extremely diverse.

In Table I, we present for each metric and each matching problem studied the number of solutions of the instance from the most succesful run, from the least succesful run, and the average of the outcomes of all runs. In Table II we compare the metrics by their runtimes, rounded down to the minute. All experiments in this section were done on the same DELL M600, with an Intel Xeon E5430 processor (2.66Ghz).

For all three matching problems considered, the contrarian score outclasses the split and Latin score. For the 2DSM problem the higher best run is obtained by increasing the number of solutions, but in average increasing the contrarian score leads to a better run. This allows us to formulate the following observation: to find a 2DSM instance with many solutions, it is better in average to increase the contrarian score than the number of solutions, even though *the number of solutions is the objective in the first place*. Furthermore, increasing the contrarian score is *80 times* faster.

For the 3DSM problem, increasing the contrarian score does not lead to more solutions than increasing the number of solutions. However, the difference between the two runtimes is even larger. The contrarian score experiment found an instance with only 5% less solutions than the best instance found by increasing the number of solutions, with the former experiment being *340 times* faster.

The 3DSM$_{strong}$ problem seems to be the one where the contrarian score performance is the weakest, which mirrors what we observed in the previous section. Note however that the contrarian score does have the higher worst-case run.

To summarize, the experiments in this section show that the contrarian score is a promising tool to generate matching instances with many solutions, if only because it is much faster to compute than standard local search methods.

V. CONCLUSION

We introduced the contrarian score, a metric for stable matching that aims to determine how constrained an instance is. We empirically showed that the contrarian score is highly correlated with the number of solutions for matching instances. Consequently, an higher contrarian score indicates more relaxed constraints, while a lower score is the sign of a more constrained instance.

We compared the contrarian score against other potential tightness metrics for stable matching, and found out that it provides the strongest correlation. We also tested it for various instance sizes and even fundamentally different matching problems. Each time the contrarian score proved to be a valid measure of constrainedness for the problem considered.

REFERENCES

[1] Arthur T. Benjamin, Cherlyn Converse, and Henry A. Krieger. How do I marry thee? Let me count the ways! *Discrete Applied Mathematics*, 59(3):285–292, 1995.

[2] Péter Biró and Eric McDermid. Three-sided stable matchings with cyclic preferences. *Algorithmica*, 58(1):5–18, 2010.

[3] Lin Cui and Weijia Jia. Cyclic stable matching for three-sided networking services. *Computer Networks*, 57(1):351–363, 2013.

[4] Daniel Eilers. Irvine compiler corporation technical report. *ICC TR1999-2*, 1999.

[5] Kimmo Eriksson, Jonas Sjöstrand, and Pontus Strimling. Three-dimensional stable matching with cyclic preferences. *Mathematical Social Sciences*, 52(1):77–87, 2006.

[6] Guillaume Escamocher and Barry O'Sullivan. Three-dimensional matching instances are rich in stable matchings. *CPAIOR 2018*.

[7] Guillaume Escamocher, Mohamed Siala, and Barry O'Sullivan. From backdoor key to backdoor completability: improving a known measure of hardness for the satisfiable CSP. *CPAIOR 2018*.

[8] David Gale and Lloyd S Shapley. College admissions and the stability of marriage. *The American Mathematical Monthly*, 69(1):9–15, 1962.

[9] Ian P. Gent, Ewan MacIntyre, Patrick Prosser, and Toby Walsh. The constrainedness of search. *AAAI 1996*.

[10] Carla P. Gomes, Cèsar Fernández, Bart Selman, and Christian Bessière. Statistical regimes across constrainedness regions. *Constraints*, 10(4):317–337, 2005.

[11] Chien-Chung Huang. Circular stable matching and 3-way kidney transplant. *Algorithmica*, 58(1):137–150, 2010.

[12] Robert W. Irving and Paul Leather. The complexity of counting stable marriages. *SIAM J. Comput.*, 15(3):655–667, 1986.

[13] Robert W. Irving, David Manlove, and Sandy Scott. The stable marriage problem with master preference lists. *Discrete Applied Mathematics*, 156(15):2959–2977, 2008.

[14] David F. Manlove. *Algorithmics of Matching Under Preferences*, volume 2 of *Series on Theoretical Computer Science*. 2013.

[15] Yongshao Ruan, Henry A. Kautz, and Eric Horvitz. The backdoor key: A path to understanding problem hardness. *AAAI 2004*.

[16] Tian Sang, Fahiem Bacchus, Paul Beame, Henry A. Kautz, and Toniann Pitassi. Combining component caching and clause learning for effective model counting. *SAT 2004*.

[17] Harborne W. Stuart, Jr. The supplier—firm—buyer game and its m-sided generalization. *Mathematical Social Sciences*, 34(1):21 – 27, 1997.

[18] Edward G. Thurber. Concerning the maximum number of stable matchings in the stable marriage problem. *Discrete Mathematics*, 248(1-3):195–219, 2002.

[19] Ryan Williams, Carla P. Gomes, and Bart Selman. Backdoors to typical case complexity. *IJCAI 2003*.