Proceedings of TMCE 2016, May 9-13, 2016, Aix-en-Provence, France, edited by I. Horváth, J.-P Pernot, Z. Rusák.

MATCHING DESIGNERS AND 3D PRINTING SERVICE PROVIDERS USING GALE-SHAPLEY ALGORITHM

Naman Mandhan
Joseph Thekinen
Alan Lo
Jitesh H. Panchal
School of Mechanical Engineering
Purdue University
{nmandhan,jthekine,alo,panchal}@purdue.edu

ABSTRACT

With the increasing availability of mid-to-low price 3D printers, it is increasingly possible for individuals and medium-sized enterprises to own such machines. However, such owners rarely utilize the full capacity of these machines. The excess capacity can be made available to interested designers who would like to get their designs printed, but do not own the machines. This has resulted in an emergence of online portals, where machine owners can register and advertise their printing resources, and designers can avail these resources to choose the machine that best suits their design. Presently a first-come-first-serve approach is used to match the designers with machine owners. The primary limitations of this approach are that (a) the capacity of the machines is highly under-utilized and (b) the matching is based solely on the designers' preferences while ignoring the machine owners' preferences. To address these limitations, we propose the use of Gale-Shapley matching algorithm after applying the utility theory to obtain the designer and manufacturer preferences for one another. The use of Gale-Shapley matching algorithm is evaluated and compared with the first-come-first-serve approach. The results of the study indicate that the approach based on Gale-Shapley matching improves the total social welfare over the present first-come-first-serve approach. While this method is slightly biased in favor of either the designers or the machine owners, given the limitations of the algorithm, both sides have improved utility and get matched to a design or machine high in their preference ordering.

KEYWORDS

Matching, utility theory, 3D printing, Gale-Shapley algorithm, social utility.

1. INTRODUCTION

Recent developments in additive manufacturing have placed the manufacturing industry on the brink of a revolution. In the 2013 State of the Union address, US president Barack Obama publicly endorsed the 3D printing technology as a transformation that can change the very face of the manufacturing industry [1].

3D printing is expected to usher in the third Industrial Revolution. It offers high flexibility in the design of products. Manufacturing can now take place in the same country where products are consumed, eradicating the need to rely on intermediate manufacturing platforms [2]. Design files can now be sent and manufactured anywhere in the world [2]. It offers the ability to manufacture different products without the need for costly retooling [3].

Many more advantages of 3D printing exist, however this paper focuses on the advantages that are catalyzing the decentralization in manufacturing. Every day, designers are gaining access to 3D printing technologies more easily as 3D printers become cheaper and the applications of the technology expands. Up until recently, their applications were limited to creating prototypes for aerospace, defense and automotive companies before embarking on more expensive ventures [4]. Filton [5] quantifies this claim by stating that more than 20% of the models printed using these machines are now final products rather than prototypes and this number is expected to rise to 50% by 2020. Due to the growth in the size of this industry, there is a need for a central system to facilitate the interaction between agents such as designers and manufacturers. Such a system must be robust and immune to the manipulations and strategic behavior by self-interested interacting agents.

Additive manufacturing has helped bridge the gap between designers and manufacturers by enabling rapid transition of concepts into physical prototypes and final products. One way in which companies are able to bridge this gap is by producing affordable 3D printers that designers can use to print and market their own products. MakerBot [6] and 3D Systems [7] are examples of two such companies. Another reason for this change is that now designers have access to robust 3D modeling software such as Solidworks [8] and CATIA [9] which allow them to create and edit 3D models supported by several types of 3D printing processes.

To serve the designers for whom it is economically not viable to own different printers for their needs, there has been an emergence of *centralized* service organizations, such as Shapeways [10], who own a variety of 3D printers. Designers can submit geometric models and get them printed. These companies typically also offer quality checks and assistance to designers to help them market and sell their products in return for a commission to the company. Moreover, companies such as Shapeways and iMaterialise [11] also provide avenues for people with ideas to connect with professional designers who can help them develop their designs [12].

In addition, an alternate, *decentralized* scenario exists where designers who do not possess the necessary resources to make physical prototypes of their designs are able to connect with individual agents who do own those resources. These interactions are facilitated by service organizations such as 3DHubs [13], where designers upload their 3D models to an online platform and are able to pick a machine that they would like to get their part printed from. The machine owners complete these 3D printing tasks for a price decided based on the requirements of the designers for their submitted designs. The machines range from desktop printers to industrial level 3D printers, giving designers a myriad of options to choose from based on their needs.

Given the immense boom in 3D printing services that are increasingly decentralizing the design process, there exists a need for a framework that can accommodate the preferences of all interacting participants in these services while efficiently allocating 3D printing tasks. So far this problem was approached on a First-Come-First-Serve (FCFS) basis. Though inefficient, FCFS became popular primarily due to its simplicity. Hence we address the research question, "*How can the matching in decentralized design and manufacturing be improved over the present First-Come-First-Serve (FCFS) approach*?" The central hypothesis is that Gale-Shapley (GS) based matching offers improvement over the FCFS approach with regards to utility gained by both designers and manufacturers in a decentralized design and manufacturing framework.

Matching entities in a bipartite setting is a well-studied topic and several algorithms have been developed. These algorithms have been broadly developed for three scenarios – (a) neither set of agents consists of decision makers, (b) only one set of agents make decisions, and (c) both sets of agents make decisions. Decentralized design and manufacturing falls in the third category where both designers and manufacturers are utility maximizing agents. An appropriate algorithm in this setting should be able to consider the designer and manufacturer utilities simultaneously, and should also be immune to their strategic behavior. For example, if the Hungarian Generalized Assignment algorithm [14] is used to allocate the manufacturing resources to designers then it may fail to satisfy both the manufacturers and the designers simultaneously. Gale-Shapley is a classic algorithm that has been used for bipartite matching in the last 50 years in applications such as matching students to colleges, medical residents to hospitals. Hence, we propose a systematic approach using Gale-Shapley deferred acceptance algorithm [15] to efficiently handle this decentralized setting.

2. SOLUTION METHODOLOGY

2.1. Problem Formulation

This paper focuses only on the decentralized scenario. This scenario can be generalized as a set of m service seekers, $D = \{d_1, d_2 ... d_m\}$ and a set of n service providers, $S = \{s_1, s_2 ... s_n\}$ independently trying to maximize their individual payoffs. The service seekers and service providers will be collectively referred to as agents, denoted by set $A = \{D, S\}$. All agents of set S are alternatives for agents in set D and vice-versa.

Associated with each agent is an exhaustive[1] list of strictly ordered preference ordering for each alternative of the opposite set. The preference

[1] Exhaustive means the preference list should contain all the members of the opposite side.

ordering for an agent is generated by utilizing utility-based decision theory. The agents reveal information about their characteristic attributes which are usually probability distributions over the possible values that the attributes can take. All agents of the set D value certain set of attributes of their alternatives in set S, which are called the 'service provider attributes', and are denoted as $SA = \{SA_1, SA_2 \ldots SA_x\}$. Similarly, agents of S value 'service seeker attributes' viz. $DA = \{DA_1, DA_2 \ldots DA_y\}$.

While generating the preference ordering for alternatives in S, d_is are the decision makers and they reveal their qualitative preference characteristics for each attribute in the set SA. They also specify their significance level of each attribute, which defines how important the attribute is to the decision maker in question, and is quantified as the weight for that attribute. Based on the qualitative preferences to attributes, probability distribution of attributes, significance level of attributes and dependency among attributes, multi-attribute expected utility is calculated for each alternative s_i by each service seeker d_j, denoted by $EU_d(i, j)$. This is further repeated by each s_i as a decision maker. Similarly, expected utility is calculated for alternative d_i by service provider s_j, denoted by $EU_s(i, j)$. The preference ordering of each agent is generated based on the notion that the higher the expected utility is, the higher its rank is in the preference list.

A matching M is a one-to-one correspondence between the elements of sets D and S. If designer d and service provider s are matched in M, we call the ordered pair (d, s) a matching pair and we set $s = p_M(d)$, i.e., s is the matching partner of d under M or if vice-versa, we can set $d = p_M(s)$. If $p_M(d_i)$ is the r^{th} element in order in the preference ordering of d_i then the rank of d in M is r_{di}. The average rank of D is defined as $r_D = \frac{\sum_i r_{di}}{m}$ and the average rank of S is defined as $r_S = \frac{\sum_i r_{si}}{m}$.

In the following, we define optimality and stability – two key properties of a matching M.

Optimality: A matching M is said to be optimal if the weighted average of r_D and r_S is as low as possible. M is said to be $D-$optimal if r_D is significantly lower than r_S and is said to be biased in favor of D. Similarly, $S-$optimality and bias in favor of S is defined. An alternate definition of optimality [16] is in terms of the average expected utility of each set i.e., r_D and r_S are replaced by EU_D and EU_S respectively; with $EU_D = \frac{\sum_i EU_d(i, p_M(d_i))}{m}$ and $EU_M = \frac{\sum_i EU_s(i, p_M(s_i))}{m}$ with the exception that here a higher average expected utility suggests greater optimality. Both these definitions are identical when the agents truthfully reveal their preferences. If the matching M incentivizes falsified preference revelation, then higher rank does not imply higher expected utility.

Stability: M is said to be a stable matching if there are no blocking pairs. A designer d and service provider s are said to be a blocking pair if the following conditions hold true: (i) d and s are not partners in M; (ii) d prefers s to $p_M(d)$ and s prefers d to $p_M(s)$. An alternate definition of stable matching is that it is a matching with 'justified envy' i.e., even though an agent x envies an alternative y different from his partner in M, y will not prefer x to his partner in M and hence both y and x will not break out of M [16]. Thus, all agents matched through M will submit to the matching mechanism if M is a stable-matching mechanism.

The objective is to achieve the right tradeoff between stability and optimality. Maintaining stability is important in a decentralized setting else the agents will resort to strategic behavior such as providing falsified preference ordering, forming coalition strategies and ties outside the centralized platform. This indirectly deteriorates the matching efficiency and may lead to unfair outcomes [16]. At the same time, ensuring stability in the final matched outcome comes at the cost of optimality.

We model this decentralized setting as a matching problem ensuring maximum optimality without compromising on stability. The agents submit their preference characteristics and the central platform performs the matching.

2.2. Proposed Approach to Applying Matching in Decentralized Design and Manufacturing Settings

Figure 1 summarizes the steps adopted in solving the problem. These steps are discussed next.

Phase 1: Data Collection

This phase begins with collecting data of the service providers, D and service seekers, S. For example, m independent designers form the set D; while n independent machine-owners form the set S. Each designer submits a design along with his/her preferences for machine attributes, SA, such as resolution, material characteristics, size requirements. Each machine-owner submits his/her preferences for design attributes, DA, such as resolution requirements, complexity, urgency, etc. We assume that information about attributes of all agents in D and S viz. DA and SA is common knowledge. These attributes are of three types: (a) unique to $d_i \in D$, (b) unique to $s_i \in S$, and (c) unique to each service provider-seeker ordered pair $a_{i,j} \in D \otimes S$. The amount of time required for each design is unique to each designer-machine pair and would fall in category c, whereas other distinct machine characteristics such as resolution capabilities belong to category b.

Phase 2: Utility-Based Preference Ordering

In this phase, we mathematically formulate the attribute preferences using expected utilities of each agent for each alternative. These expected utilities are in turn used to calculate the strict preference ordering that serves as the input for the matching process. In order to calculate the expected utilities the following steps are followed:

Step 1: Attribute Probability Distributions and Utility Functions

The first step is to define the probability distributions for each attribute. This is done based on the information provided by the set of agents, A. For example, consider the tensile strength (TS) of the material as an attribute of the service seeker. Based on experiments and historical data the variability in TS is quantified as a probability distribution over a range of values it assumes.

Next, utility functions of the agents towards the attributes characterizing their alternatives are defined. The evaluation of these utility functions is carried out using the lottery-question approach, as illustrated for 3D printing by Fernandez et al. [17].

Step 2: Attribute Weights and Dependencies

Attribute weights and dependencies are important to obtain the multi-attribute utility function. Ideally, considering the generalized case of dependency and tradeoffs between attributes, the multi-attribute utility function u_i for agent d_i in D is $u_i(SA_1,\ldots,SA_x) = f\left[f_1(SA_1),\ldots,f_x(SA_x)\right]$ for $i = 1,\ldots,m$. Similarly, utility function u_j for agent s_j in S is $u_j(DA_1,\ldots,DA_y) = f\left[f_1(DA_1),\ldots,f_x(DA_y)\right]$ for $i = 1,\ldots,n$. The above equation can be simplified by applying certain independence assumptions in designer preferences. These assumptions are established based on:

1. *Additive independence of attributes:* There are two independence conditions viz. utility independence and additive independence. The methods for defining and testing the same are detailed by Keeney and Raiffa [18] and Thurston [19].
2. *Evaluating the scaling constants:* The scaling constants define the significance of the attribute for the agent.

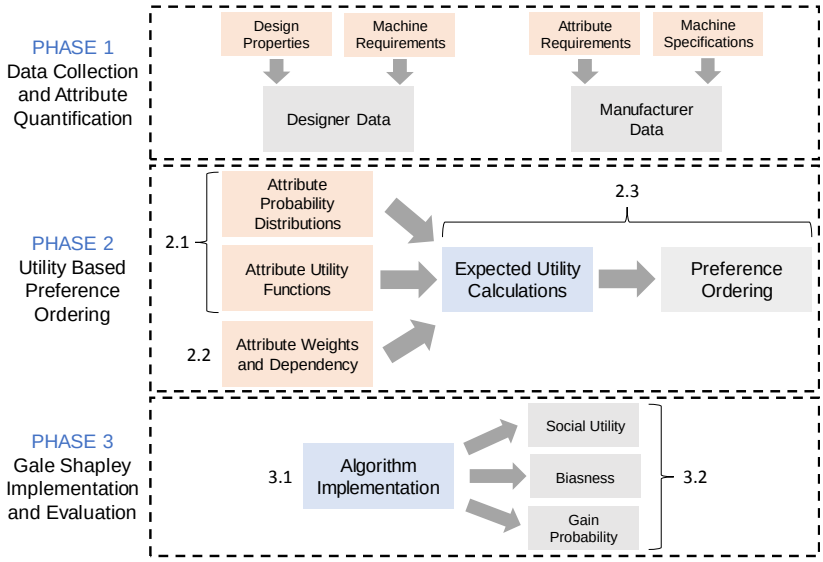

Figure 1 Flowchart of the proposed approach

Step 3: Expected Utility Calculations and Preference Ordering

The single-attribute utilities of the agents, along with the probability distribution for each attribute, and scaling constants are combined to generate the overall expected utilities that each agent has for each of the alternatives. The alternatives are then ranked by each agent based on decreasing order of their expected utilities. Thus, a preference ordering of each agent is generated. For example, if an agent in *D* has preferences ordering for every alternative in set *S*. This preference list is exhaustive and strict in ordering (no ties are allowed).

Phase 3: Matching Algorithm Implementation and Analysis

Once the preference ordering is generated for each agent, we choose an appropriate matching algorithm. Examples of matching algorithms include Gale-Shapley deferred acceptance algorithm [15], and Top-Trading Cycle algorithm [20]. The average expected utility and rank achieved by each agent are then evaluated to analyze the performance of the algorithm. The goals in matching are to maximize the average expected utility (and rank), and to minimize the bias ensuring stability.

2.3. An Illustrative Example

Consider an illustrative setting were 10 designers are trying to get their designs printed through 10 machine-owners ($m = n = 10$). The details of the designs are provided in Table A1. The designers base their preferences for machines on three machine-owner attributes viz. resolution (SA_1), tensile stress (SA_2) and size (SA_3). Machine-owners choose the design based on three design-attributes: printing time (DA_1), build area consumed (DA_2), and resolution (DA_3).

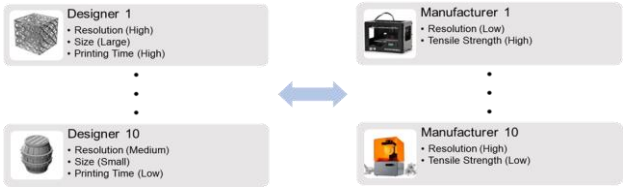

Figure 2 Designer and manufacturer matching scenario with attributes

In Figure 2, each block shows the attributes corresponding to that agent. The qualitative preferences of the agents towards the attributes are indicated in parentheses. The preferences of each agent are independent of the preferences of other agents. In the rest of this section, we illustrate the application of the proposed approach for this example.

Phase 1: Data Collection

Ten designs from 'Thingiverse' [21], each with a different volume and base area are downloaded (see Table A1). Five different machines are chosen. The printing time of each design in all the five machines is calculated using appropriate software for each machine. Among the five machines chosen there are three Fusion Deposition Modeling (FDM) machines and two Stereolithography (SLA) machines. Sets of FDM and SLA materials are selected and their material properties are obtained from 'Materialise' [22]. The machine type and the material sets are then permuted[2] to obtain ten unique machine owners (see Table A2) who form members of the set S .

Designer Preferences

The designers' preferences are based on the machine-owner attributes. The first machine-owner attribute (SA_1) is **resolution**. This is an important attribute because the quality, detail capability, finish and accuracy of the final printed part depend on SA_1. Designers are given the option to choose either a low, medium or high resolution. The resolution capabilities of the five machines short-listed vary from 100 microns to 5 microns. This range is segregated into high, medium and low resolution regimes (see Table A3). Their individual utility functions are then formulated based on the category they fall into.

SA_2 is **tensile strength** (TS). Based on the desired application, the designer may prefer a material with low, medium or high TS. Material data, collected from the Materialise website [22] for different FDM and SLA materials show that TS ranges from 22 MPa to 72 MPa for the chosen materials. TS is divided into low, medium and high regimes.

SA_3 (**size**) is a mandatory requirement because violating these restraints make printing infeasible[3]. If size of the design is beyond the capacity of the

[2]It was assumed that the FDM materials could be used only in a FDM machine, and similarly for SLA.

[3] It is assumed that the products are printed as a single piece without assembling and hence the size of the design must be within the capacity of the machine, else the part cannot be printed.

machine then the expected utility for the designer to get matched with the machine-owner in consideration is zero[4].

Manufacturer Preferences

The machine-owner preferences are based on the designer attributes such as printing time, resolution, geometric properties.

DA_1 (**resolution)**, is important to the machine-owner due to two major reasons:

1) *Economic reason*: the machine-owners prefer to fully utilize their resolution capabilities, e.g., consider a high end machine that can print resolution as low as 5 microns – the machine-owner prefers to print a high resolution product.
2) *Strategic reason*: machine-owner may prefer to print other products in the same run and would consider a particular resolution range as an ideal match. In this illustrative scenario, the strategic reason is ignored as it is one-to-one matching and machine-owner utility is quantified such that the higher the resolution capacity utilized, the higher the utility.

DA_2 (**size)** is quantified based on the percentage of the machine build area consumed. This is unique to each design-machine pair. Due to economic reasons, suggested earlier for resolution, it is assumed that machine-owners have higher utility for higher percentage area consumption. This may not always be true. For example, a reduced print area means several products can be printed in the same job thus saving up on set-up time and improving overall efficiency. Such concerns can be modeled by re-defining the utility function. The proposed solution is still valid.

DA_3 (**printing time**) is also unique for each machine-design pair because it depends on the complexity of the design and type of printer employed to perform the task[5]. In this analysis, setting up and post-processing time are excluded while calculating DA_3.

[4]Only designs that satisfy the mandatory attributes in all ten manufacturers are selected. This was done to ensure that the preference list was exhaustive.

[5]The actual time may also depend on the resolution requirement of the designer rendering dependency between two attributes. For simplicity, it is assumed that all the attributes are independent of one another. Such concerns can be addressed in the utility calculation and do not lie within the primary objectives of the paper.

Phase 2: Utility Based Preference Ordering

Step1: Attribute Probability Distributions and Attribute Utility Functions

The simplest case of a uniform probability distribution between identified upper and lower bounds for each attribute is assumed. Then, the utility function for each designer and manufacturer attribute is defined. For brevity, only SA_1 is discussed. Similar procedures are followed for other attributes and the numerical results are briefed in Table A3 in the Appendix. As mentioned earlier, there are high, medium and low resolution requirements. The designer who prefers high resolution has a maximum utility (=1) for 5 microns, minimum utility (=0) for 22 microns and 0.6 utility[6] for 13.5 (mean of 5 and 22). A second-order polynomial fitting the above three points is used to generate the utility function. The utility is 0 for all values of SA_1 above 22 microns and below 5 microns. For designers who prefer medium resolution there is an ideal intermediate resolution, above and below which utility drops from the optimal value (=1) to zero at both lower unacceptable and upper unacceptable values. Two polynomials are used to define the left and right hand-side utility. Low resolution is similar to high resolution, except that 300 microns has utility 1 and 90 microns has utility 0.

Step 2: Attribute Weights and Dependency

Additive independence of attributes: For this example, multi-utility independence and the additive independence property are assumed. Hence $DU_i = \sum_{j=1}^{x} k_{ij} du_{i,j}\left(SA_j\right)$ is the multi-attribute utility function of designer d_i; and $SU_i = \sum_{j=1}^{y} k_{ij} su_{i,j}\left(DA_j\right)$ is the multi-attribute utility function of machine-owner s_i.

Evaluation of scaling constants: The scaling constant for each attribute is quantified on a scale from 1 to 5 based on how significant that attribute is to the agent. For example, the agents assign a scale 5 to those attributes that are mandatory and a scale 1 to those that are least significant. These scales are then normalized to unity.

[6]Now 0.6 was chosen to account for the risk-averse nature of the designer. All agents are assumed to be risk-averse in this analysis.

Step 3: Expected Utility Calculations and Preference Ordering

The single-attribute utilities of the designers, along with the probability distributions for each attribute and the attribute weights (*k*-values) are combined to generate the total expected utilities that each agent has for every alternative in the opposite set.

$DEU_{i,j} = \sum_{l=1}^{x} k_{i,l} \int_{slb_{i,j}}^{sub_{i,j}} du_{i,l}(SA_l)\, dp_j(SA_l)\, dy$ is the expected utility of the i^{th} designer for j^{th} machine-owner.

$SEU_{i,j} = \sum_{l=1}^{y} k_{i,l} \int_{dlb_{i,j}}^{dub_{i,j}} su_{i,l}(DA_l)\, sp_j(DA_l)\, dy$ isthe expected utility of the i^{th} machine-owner for j^{th} designer.

Here, $dp_j(SA_l)$ is the probability distribution of attribute SA_l for alternative j. sp_j is same for attribute DA_j. $sub_{i,j}$ and $slb_{i,j}$ are the upper and lower bounds respectively for attribute l of machine-owner j. $dub_{i,j}$ and $dlb_{i,j}$ are the upper and lower bounds respectively for attribute l of designer j.

The preference ordering of each agent is generated based on the rationale that the higher the expected utility for an alternative to an agent then the higher is the rank of the alternative. The preference orderings thus generated are inputs to the Gale-Shapley deferred acceptance matching algorithm.

Phase 3: Gale-Shapley deferred acceptance algorithm implementation

The Gale-Shapley algorithm (as described in [15]) is implemented with designers as proposers:

- *Initiation*: The status of each agent is set to be "unmatched".
- *Loop:*
  - While a designer d_i is unmatched
    - Select the first machine-owner s_j on d_i's list to whom m_i has not yet proposed[7]
    - If s_j is free or s_j prefers d_i to her currently matched designer then assign d_i to s_j; else s_j rejects d_i and d_i status continues to be unmatched
  - Repeat the above step for every unmatched designer d_i
- *Terminate* when all designers are either matched or run out of choices.

3. RESULTS

The solutions obtained from this algorithm are biased to the proposing side (the designers in this case). Hence, the solution lacks fairness in that it optimizes the preference structure of one side as compared to the other. Nevertheless, the solutions obtained are strategy-proof and stable. More importantly, the solution Pareto dominates any other matching mechanism that produces stable outcome. However, stability comes at the cost of optimality. Despite this compromise on optimality we show that the proposed approach offers significant improvement upon the currently used first-come-first-serve approach.

The section is structured as follows. For a given preference ordering and expected utility for each designer and manufacturer, the optimality offered by Gale-Shapley algorithm is first compared with first-come-first-serve in Section 3.1. In Section 3.2, the same analysis is repeated for several such combinations of preference ordering.

3.1. First-Come-First-Serve vs. Gale Shapley

To compare the effectiveness of the proposed approach, a set of hundred instances of the first-come-first-serve approach is compared with the results from the Gale-Shapley (GS) approach. Given that the preference ordering of the designers and machine owners does not change, GS always yields one stable solution for all hundred instances of the first-come-first-serve method, which is a property of the algorithm itself.

Consider Figure 3, where the average rank obtained by the hundred matching instances of the first-come-first-serve approach is compared with that of GS. In this case, 'rank' is defined as being the preference number to which the designer is matched. For example, if the designer has a preference ordering of 1, 2, 3…10 and is matched to 3 then he/she will attain a rank of 3. In an ideal matching scenario the average rank amongst the ten designers should be as low as possible. This is repeated for all designers (Figure 3(a)) and machine owners (Figure 3(b)).

The average rank obtained through GS is 3.4 for the designers and 4.8 for the machine owners. From Figure 3(a), it is observed that each of the hundred randomly chosen matching instances in FCFS yields

[7] In this paper, *m* are designers, considered to be the dominant side, and *w* are the machine owners.

a higher average rank than GS. In Figure 3(b), a few random matching instances are seen to perform better than GS, as shown by the data points that lie below the results of GS. This is due to the fact that GS is implemented with designers as the dominant side within this framework. Yet, in more than 85% of the cases that GS performed better.

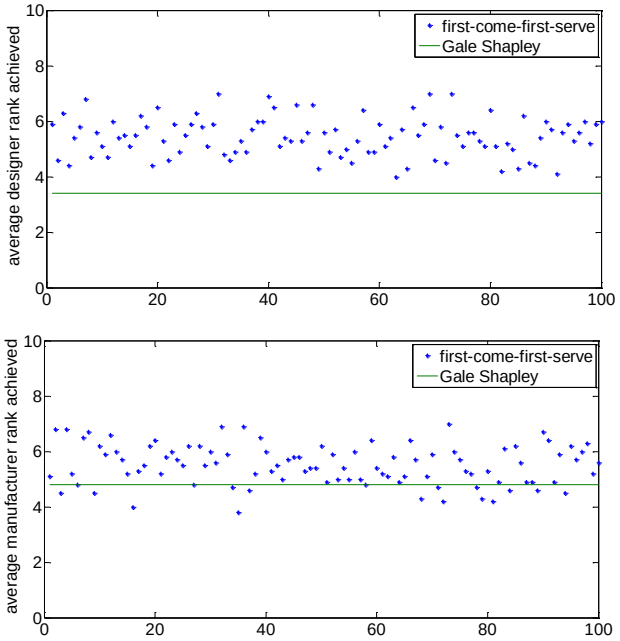

Figure 3 Average (a) designer and (b) manufacturer rank

Now consider Figure 4, where the average expected utility obtained by both methods is presented. Comparing the data reveals that the average expected utility values obtained were 0.327 and 0.187 for the designers and machine owners, respectively[8]. Counter-intuitively, the average expected utility is better than all random matching instances for the case of machine owners, however in the case of designers there are 6% of random matching cases that surpass the GS designer expected utility value. Thus, a better average rank need not always imply better expected utility and vice-versa.

In the present case there are 10! (>3.6 million) different matching instances. In each of those matching instances, the utility attained by the set of ten designers and machine owners would at most times be different, resulting in a social utility and fairness different from that of GS. GS does not yield the most optimal social utility. Amongst the stable matching algorithms, the GS algorithm (with designers as the dominant side) yields the most optimal match for designers and least optimal match for machine owners. The results obtained from analyzing the average rank and utility suggest that GS with designers as the dominant side is marginally biased in favor of the designers, as is to be expected, given the workings of the algorithm itself. However, the situation is not that bad as it sounds for machine owners. It is shown that most of the random matching (achieved by FCFS) yields social utility much lower than the one obtained through GS even in the case of machine owners. This is because it is only a tiny proportion of stable matching instances that the social utility is least optimal for machine owners.

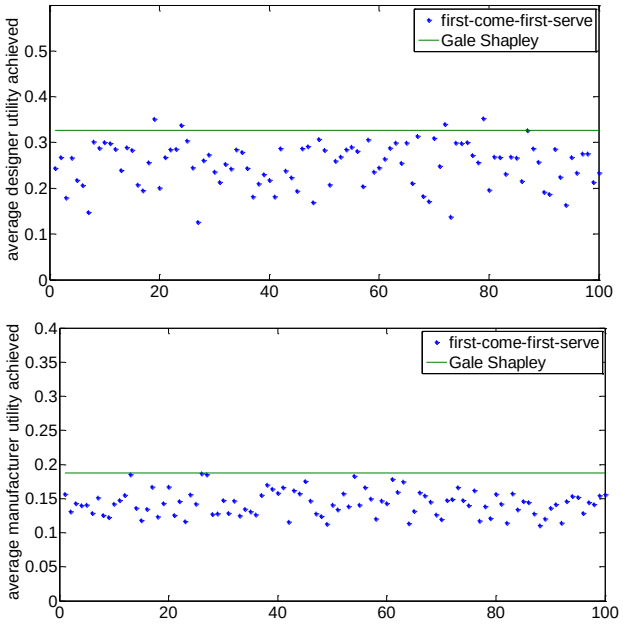

Figure 4 Average (a) designer and (b) manufacturer utility

3.2. Repeating with Different Initial Conditions

The preference ordering and hence the matching depends on the expected utilities of designers and machine owners for one another. These expected utilities depend on the preferences of designers and machine owners towards the attributes which has an uncertainty factor associated with it. This uncertainty is accounted for in the analysis by running the analysis for different utility functions. These different utility functions thus generated serve as the initial conditions for that iteration. The analysis is repeated for fifteen different initial conditions to

[8]A higher average does not automatically imply higher average expected utility. For example, if the preference ordering and expected utility of each manufacturer does not vary much from designer to designer then the utility gain by GS may not be as evident.

further assess the effectiveness of the proposed approach.

The results for the fifteen different initial conditions, comparing the GS and first-come-first-serve method, are shown in Figures5 and 6. The average designer rank of GS with first-come-first-serve for the fifteen cases is compared in Figure 5(a). The i^{th} box-plot summarizes the average designer rank data distribution for the hundred random matching instances in the i^{th} initial condition run. The same data is shown in Figure 5(b) for the manufacturer.

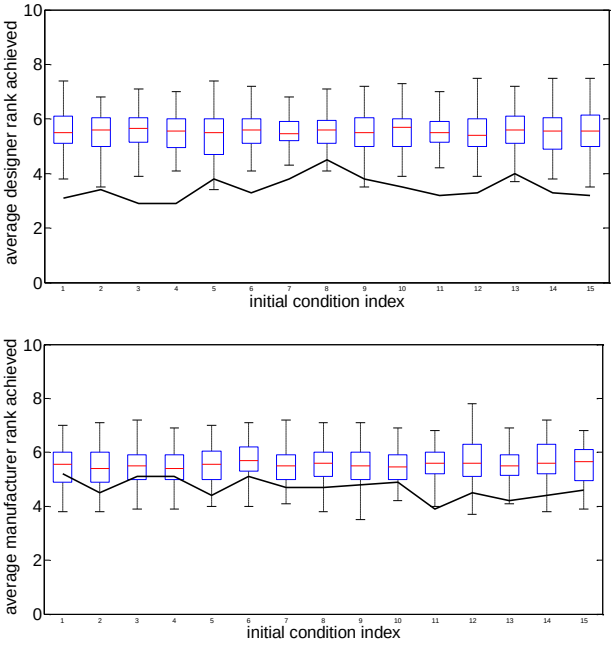

Figure 5(a) Designer and (b) manufacturer rank for different initial conditions

It is observed that the GS results lie under the average designer rank data distributions, which suggests that GS gives the better average rank, both in absolute terms and in comparison to the hundred random matching instances, for the designer than the manufacturer. Further, the results also suggest that GS gives a better rank both for designers and machine owners, despite its slight bias towards optimizing designers.

The average expected utility for each of the fifteen iterations is compared in Figure 6 for both designers (Figure 6(a)) and machine owners (Figure 6(b)).

4. CONCLUSION AND FUTURE WORK

In this paper, we implement utility maximization and the Gale-Shapley algorithm to a decentralized design scenario within the realm of additive manufacturing. The GS algorithm is employed with designers as the dominant side in the matching problem. It is established that GS, despite its bias towards designers as the dominant side, makes both designers and machine owners better off when compared to the present first-come-first-serve approach. However, GS does not yield the most optimal match for all designers and machine owners. Nevertheless, with the restriction of justified envy GS yields the most optimal match for designers. Justified envy is important as we want all designers and machine owners to use the matching framework and not go for strategies outside the platform which may be detrimental.

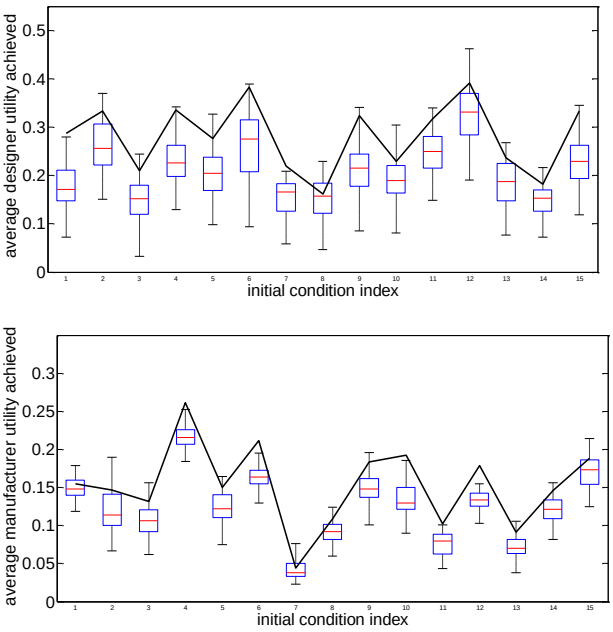

Figure 6 (a) Designer and (b) manufacturer expected utility for different initial conditions

It can be concluded that preference ordering based on expected utility maximization followed by the Gale-Shapley algorithm is an effective approach that can be followed in the decentralized design and production scenario by massive advent of additive manufacturing. It offers significantly higher advantage over the present first-come-first-serve approach.

In the present analysis, the scenario is restricted to be a stable matching problem. This is important in the scenarios where both designers and machine owners are independently trying to strategize and optimize the match they receive. However, in those settings where strategic behavior is not important GS is not the most suited matching algorithm. For example,

consider a design company where the designers are availing the 3Dprinting resource of the company to test their prototypes. Here the concept of stable matching would be impractical. Other matching algorithms like the Top Trading Cycle [20] and Hungarian algorithm [14] would prove more effective.

In addition, the scope of this paper also restricts the scenario to one-to-one matching. Theoretically, higher efficiency can be achieved by matching multiple designs to the same manufacturer until their schedule is completely utilized. Matching algorithms that relax such assumptions are expected to improve the results further. Moreover, in the same or additional scenarios, more attributes such as the cost of the part, flexural modulus and the geographical distance between agents can be considered for expected utility calculations.

ACKNOWLEDGMENTS

The authors gratefully acknowledge financial support from the US National Science Foundation through NSF CMMI grants 1265622 and 1329979.

REFERENCES

[1] Gross, D., (2013), "Obama's Speech Highlights Rise of 3-D Printing", Website, http://www.cnn.com/2013/02/13/tech/innovation/obama-3d- printing/.

[2] Campbell, T., Williams, C., Ivanova, O., Garrett, B., (2011), "Could 3D Printing Change the World?", Atlantic Council, Washington, D.C.

[3] Rao, R. V., and Padmanabhan, K. K., (2007), "Rapid Prototyping Process Selection Using Graph Theory and Matrix Approach", Journal of Materials Processing Technology, Vol. 194, No. 1-3, pp. 81-88.

[4] Royte, E., (2013), "What Lies Ahead for 3-D Printing?", Magazine, http://www.smithsonianmag.com/science-nature/what-lies-ahead-for-3-d-printing-37498558/?no-ist

[5] Filton, (2011), "The Printed World", Magazine, http:// www.economist.com/node/18114221.

[6] "Makerbot", Website, http://www.makerbot.com/

[7] "3D Systems", Website, http://www.3dsystems.com/

[8] "Solidworks", Website, http://www.solidworks.com/

[9] "CATIA", Website, http://www.3ds.com/products-services/catia

[10] "Shapeways", Website,https://www.shapeways.com/

[11] "iMateralise", Website,http://i.materialise.com/

[12] D' Aveni, R. A., (2013), "3-D Printing Will Change the World", Magazine, https://hbr.org/2013/03/3-d-printing-will-change-the-world/

[13]"3D Hubs", Website,https://www.3dhubs.com/

[14] Kuhn, H. W., (1955), "The Hungarian Method for the Assignment Problem", Naval Research Logistics Quarterly, Vol. 52, No. 1, pp. 7-21.

[15] Gale, D., and Shapley, L. S., (1962), "College Admissions and the Stability of Marriage", The American Mathematical Monthly, Vol. 69, No.1,pp. 9-15

[16] Gusfield D., Irving R. W., (1989), "The Stable Marriage Problem: Structure and Algorithms", MIT Press, Cambridge, MA.

[17] Fernández, M. G., Seepersad, C. C., Rosen, D. W., Allen, J. K., and Mistree, F., (2005), "Decision Support in Concurrent Engineering - The Utility-Based Selection Decision Support Problem", Concurrent Engineering, Vol. 13, No. 1, pp. 13-27.

[18] Keeney, R.L. and Raiffa, H. (1976). Decisions withMultiple Objectives: Preferences and Value Tradeoffs, New York: John Wiley and Sons.

[19] Thurston, D.L. (1999). Real and Perceived Limitations toDecision Based Design, In: ASME DETC, Design Theoryand Methodology, Las Vegas, NV, USA, Paper Number:DETC99/DTM-8750.

[20] Shapley, L., and Scarf, H., (1974), "On Cores and Indivisibility", Journal of Mathematical Economics, Vol 1, No. 1, pp. 23-37.

[21] "Thingiverse", Website, https://www.thingiverse.com/

[22] "Technologies and Materials", Materialise, Website, http://manufacturing.materialise.com/all-available-materials

APPENDIX

Table A1. Characteristics of designers

#	Design	Makerbot	Ultimaker	Witbox	B9Creator	Form1+	Res.	TS	Area (cm^2)
D1	CarabinerClip	2.25 h	1.55 h	1.88 h	3.68 h	2.25 h	High	Low	4.28
D2	Celtic Butterfly	0.28 h	0.35 h	0.40 h	0.12 h	0.17 h	Med.	Low	23.31
D3	Dalpek Gyro Air	6.25 h	7.67 h	8.97 h	3.30 h	2.40 h	Med.	High	49.00
D4	Duo Tone Whistle	2.27 h	1.77 h	1.83 h	0.72 h	0.70 h	Low	Med.	57.5
D5	Groot Bust	4.73 h	6.15 h	6.85 h	4.27 h	2.68 h	Low	Low	38.36
D6	JFK Bust	7.73 h	6.15 h	11.15 h	5.13 h	5.33 h	Low	Low	56.91
D7	Magnetic T-Nut	0.13 h	10.13 h	0.12 h	0.25 h	0.20 h	Med.	Med.	1.32
D8	PacMan Ghost	0.77 h	0.08 h	1.07 h	1.20 h	0.88 h	Low	High	6.25
D9	Scripted Vase 2	9.48 h	1.37 h	13.00 h	4.65 h	6.25 h	High	High	57.3
D10	Voronoi Elephant	3.15 h	11.28 h	4.90 h	2.67 h	1.73 h	High	Low	30.98

Table A2. Characteristics of manufacturers

#	Machine	Res. (µm)	Volume (cm^3)	Area (cm^2)	Material 1 (TS in MPa)	Material 2 (TS in MPa)
M1	Makerbot (FDM)	100-300	6758.78	436.05	ABS (22)	PC-ABS (41)
M2	Makerbot(FDM)	100-300	6758.78	436.05	ABS (22)	PC-ABS (41)
M3	Witbox (FDM)	25-100	2578.13	156.25	ABS (22)	PC-ABS (41)
M4	Form 1+ (SLA)	50-300	12474.00	623.70	Poly 1500 (31)	Protogen (43.8)
M5	B9 Creator (SLA)	5-100	1597.64	78.62	Poly 1500 (31)	Protogen (43.8)
M6	Makerbot (FDM)	100-300	6758.78	436.05	PC (68)	ULTEM 9085 (72)
M7	Makerbot (FDM)	100-300	6758.78	436.05	PC (68)	ULTEM 9085 (72)
M8	Witbox (FDM)	25-100	2578.13	156.25	PC (68)	ULTEM 9085 (72)
M9	Form 1+ (SLA)	50-300	12474.00	623.70	Tusk Somos (63)	-
M10	B9 Creator (SLA)	5-100	1597.64	78.62	Tusk Somos (63)	-

Table A3. Utility functions

Attribute		Preference	Monotonicity	Left hand side utility($a+bx+cx^2$)			Right hand side utility($a+bx+cx^2$)	
				0	0.6	1	0.6	0
Manufacturer attribute	Resolution (µm)	High	Decreasing	5	13.5	22	-	-
		Medium	Ideal	22	41	60	75	90
		Low	Increasing	-	-	300	195	90
Manufacturer attribute	TS(MPa)	High	Decreasing	20	30	40	-	-
		Medium	Ideal	35	42.5	50	57.5	65
		Low	Increasing	-	-	60	67.5	75
Designer attribute	Printing time (hrs)	-	Ideal	0.5	2.25	4	7	10
Attribute		Preference	Monotonicity	Left hand side utility($a+bx+cx^2$)			Right hand side utility($a+bx+cx^2$)	
				0.3	0.7	1	0.55	0.1
Designer attribute	Resolution (res in µm)	-	Decreasing	-	-	Min res	mean res	Max res
Designer attribute	Area(ratio)	-	Increasing	0.1	0.55	1	-	-

Table A4. Expected utility calculations and preference ordering of designers

	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10
D1	0.0000 (9)	0.0049 (4)	0.0000 (7)	0.3383 (6)	0.0000 (8)	0.0399 (10)	0.0448 (2)	0.0399 (1)	0.3454 (3)	0.0070 (5)
D2	0.0000 (7)	0.7560 (9)	0.5409 (10)	0.7558 (2)	0.7527 (4)	0.0333 (5)	0.7893 (8)	0.5742 (3)	0.7616 (6)	0.7585 (1)
D3	0.0312 (4)	0.0339 (9)	0.0312 (2)	0.2063 (1)	0.0183 (3)	0.0000 (5)	0.0027 (7)	0.0000 (6)	0.1880 (8)	0.0000 (10)
D4	0.3393 (2)	0.3404 (7)	0.3404 (3)	0.0010 (8)	0.0010 (1)	0.3393 (6)	0.3404 (9)	0.3404 (10)	0.0057 (4)	0.0057 (5)
D5	0.0000 (7)	0.5670 (9)	0.4057 (10)	0.5668 (2)	0.5645 (4)	0.0499 (5)	0.6169 (8)	0.4556 (3)	0.5756 (6)	0.5733 (1)
D6	0.1950 (2)	0.1955 (3)	0.1955 (1)	0.0201 (7)	0.0201 (8)	0.1616 (6)	0.1621 (4)	0.1621 (5)	0.0005 (9)	0.0005 (10)
D7	0.0267 (4)	0.0302 (9)	0.0267 (2)	0.2573 (1)	0.0157 (3)	0.0000 (5)	0.0035 (7)	0.0000 (6)	0.2417 (8)	0.0000 (10)
D8	0.2413 (2)	0.2420 (3)	0.2420 (1)	0.0177 (7)	0.0177 (8)	0.2121 (6)	0.2127 (4)	0.2127 (5)	0.0006 (9)	0.0006 (10)
D9	0.0000 (9)	0.5670 (10)	0.4057 (2)	0.5668 (7)	0.5645 (4)	0.0000 (5)	0.5670 (3)	0.4057 (8)	0.5728 (1)	0.5705 (6)
D10	0.0260 (2)	0.5300 (4)	0.3866 (5)	0.5191 (7)	0.5170 (9)	0.0000 (10)	0.5040 (3)	0.3606 (8)	0.5038 (1)	0.5018 (6)

Table A5. Expected utility calculations and preference ordering of machine owners

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
M1	0.0000 (6)	0.0000 (4)	0.0544 (8)	0.3562 (5)	0.0697 (3)	0.3919 (9)	0.0000 (1)	0.3561 (2)	0.0089 (7)	0.0000 (10)
M2	0.0021 (6)	0.0697 (4)	0.0343 (8)	0.2353 (5)	0.1186 (10)	0.2364 (9)	0.0021 (2)	0.2351 (3)	0.0699 (1)	0.0980 (7)
M3	0.0000 (4)	0.0892 (6)	0.0054 (8)	0.4479 (10)	0.1040 (5)	0.4479 (2)	0.0000 (9)	0.4479 (3)	0.0892 (1)	0.1106 (7)
M4	0.0743 (6)	0.2030 (4)	0.0661 (8)	0.2883 (9)	0.2482 (5)	0.3411 (10)	0.0632 (2)	0.2851 (1)	0.2606 (3)	0.2033 (7)
M5	0.0000 (6)	0.1426 (9)	0.0007 (4)	0.1574 (8)	0.1427 (5)	0.2420 (10)	0.0000 (2)	0.1566 (3)	0.2147 (1)	0.1427 (7)
M6	0.0000 (6)	0.0000 (4)	0.0515 (8)	0.2241 (5)	0.0657 (3)	0.2577 (9)	0.0000 (1)	0.2238 (2)	0.0086 (7)	0.0000 (10)
M7	0.0029 (6)	0.0930 (4)	0.0351 (8)	0.3137 (5)	0.1418 (10)	0.3147 (9)	0.0029 (2)	0.3135 (3)	0.0931 (1)	0.1213 (7)
M8	0.0000 (4)	0.0780 (6)	0.0114 (8)	0.3919 (10)	0.1092 (5)	0.3919 (2)	0.0000 (9)	0.3919 (3)	0.0780 (1)	0.1230 (7)
M9	0.0624 (6)	0.1391 (9)	0.0448 (5)	0.1975 (4)	0.2161 (8)	0.2883 (10)	0.0435 (2)	0.1960 (1)	0.2365 (3)	0.1393 (7)
M10	0.0000 (6)	0.2377 (9)	0.0012 (4)	0.2623 (8)	0.2379 (5)	0.3093 (10)	0.0000 (2)	0.2610 (3)	0.2780 (1)	0.2378 (7)