Journal of Manufacturing Systems 43 (2017) 327–338

Contents lists available at ScienceDirect

Journal of Manufacturing Systems

journal homepage: www.elsevier.com/locate/jmansys

Technical Paper

Resource allocation in cloud-based design and manufacturing: A mechanism design approach

Joseph Thekinen, Jitesh H. Panchal*

School of Mechanical Engineering, Purdue University, West Lafayette, IN 47907, USA

ARTICLE INFO

Article history:
Received 10 April 2016
Received in revised form 15 July 2016
Accepted 18 August 2016
Available online 16 September 2016

Keywords:
Cyber manufacturing
Matching
Resource allocation

ABSTRACT

The focus of this paper is on matching service seekers and service providers, such as designers and machine owners, in cloud-based design and manufacturing (CBDM). In such decentralized scenarios the objectives and preferences of service seekers are different from those of service providers. Current resource configuration methods are unsuitable because they optimize the objectives of only one type of participants – either service seekers or service providers. Existing marketplaces based on first-come-first-serve (FCFS) approach are inefficient because they may not result in optimal matches. To address these limitations there is a need for mechanisms that result in optimal matching considering the private preferences of all the agents. In this paper, we formulate the resource allocation problem as a bipartite matching problem. Four bipartite matching mechanisms, namely, Deferred Acceptance (DA), Top Trading Cycle (TTC), Munkres, and FCFS are analyzed with respect to desired properties of the mechanisms such as individual rationality, stability, strategy proofness, consistency, monotonicity and Pareto efficiency. Further, the performance of these mechanisms is evaluated under different levels of resource availability through simulation studies. The appropriateness of matching mechanisms for different scenarios in CBDM such as fully decentralized, partially decentralized and totally monopolistic are assessed. Based on the analysis, we conclude that DA is the best mechanism for totally decentralized scenario, TTC is most appropriate when cloud-based resources are used in an organizational scenario, and Munkres is the best mechanism when all resources are owned by a single agent.

1. Introduction: matching decentralized service seekers and providers

Cloud-based design and manufacturing (CBDM) is a decentralized, service-oriented design and manufacturing model where participants utilize product development resources, such as CAE tools and manufacturing equipment, using cloud computing, and other related technologies [30]. It is an extension of cloud-based manufacturing (CBM), which is a model for "ubiquitous, convenient, on-demand network access to a shared pool of configurable design and manufacturing resources" [31]. One of the primary advantages of the decentralized model of design and manufacturing over the traditional manufacturing model of owning all resources is that it addresses the contradiction between *scarcity* and *redundancy* of manufacturing resources [25]. It allows small and medium-sized enterprises (SMEs) who lack manufacturing equipment to benefit from enterprises who own the equipment, but do not use them at the full capacity. Such a decentralized model is particularly attractive with the availability of Internet-connected digital manufacturing equipment such as 3D printers. In such an environment, designers can print their designs at any 3D printer connected to the cloud rather than at one particular site.

CBDM involves interactions among two groups of participants: service seekers and service providers. *Service seekers* need to manufacture or use computational resources, but do not possess the capabilities to do so. *Service providers* own and operate equipment or other resources and are ready to offer users instantaneous access to these capabilities. An effective CBDM platform must be able to effectively support the important tasks of resource discovery, service scheduling, service matching. Several research efforts have been focused on issues such as resource virtualization technologies, resource and service publication and discovery [24], service composition, efficiency [26], reliability and security management [27]. A review of challenges and research gaps in these emerging manufacturing models is provided by Tao and co-authors [23].

The focus in this paper is on service matching, which involves determining which service providers will serve different service seekers. The service seekers and the service providers have

* Corresponding author.
E-mail address: panchal@purdue.edu (J.H. Panchal).

http://dx.doi.org/10.1016/j.jmsy.2016.08.005
0278-6125/

Nomenclature

$\|P\|$	number of service providers
$\|S\|$	number of service seekers
μ	matching mechanism
$\mu(d_i)$	agent to which d_i is assigned through μ
μ_R	matching mechanism μ for the preference ordering R
f	single attribute utility
$j \succ_i k$	i strictly prefers j over k
$j \succeq_i k$	i prefers j over k
P	set of service providers
p	probability distribution of the attribute values
p_{-i}	set of all service providers excluding p_i
q_{p_j}	vacancies offered by service provider p_j
R_{p_j}	preference ordering of provider p_j over its alternatives
R_{s_i}	preference ordering of seeker s_i over its alternatives
S	set of service seekers
s_{-i}	set of all service seekers excluding s_i
u_{ij}	utility of agent i being matched to alternative j
w_{ki}	weight for attribute X_k for agent i
X	set of all attributes

different, often conflicting, objectives. Service seekers are interested in desired part quality at a minimum price, while service providers are generally interested in maximizing revenue from their available capacity. Additionally, the specific quality desired by each service seeker, and the price they are willing to pay are different; and the capabilities of each service provider may be different. Therefore, there are as many different objectives as there are participants in the system. One of the primary requirements of a CBDM platform is to determine an *optimal allocation of resources considering the objectives of all the participants*.

Conventional resource allocation methods based on multi-objective optimization are inappropriate for matching resources to service seekers in decentralized scenarios because they optimize the objective of one party only. The commonly used approach for matching in decentralized scenarios is a *marketplace* where service providers display capabilities and prices at a central location (e.g., on a website), and the service seekers self-select the providers based on their needs. This is a first-come first-serve approach (FCFS) because if a service provider's resource is available, it can be used by the service seeker who requests it first, and is willing to pay the asking price. Such a model is adopted in 3Dhubs [1], an online 3D printing service platform with around 25,000 service providers.

Several inefficiencies arise when service seekers are matched to service providers using FCFS. First, when resources are scarce, service seekers may have to wait indefinitely in the queue for the most sought after service providers. Second, only the service seeker's preferences are considered here, the preferences of the service provider are not explicitly considered. For example, a service provider with a high resolution 3D printer, which is more suitable to print jobs that demand higher detail capability, may be chosen first by a seeker who does not need such capability. Third, even in cases where two service seekers have first preference for the same service provider, the utility that each service seeker gains may be significantly different. Therefore, the match obtained from FCFS may not be optimal from the standpoint of the entire set of participants. Fourth, it is possible for participants to try and "game" the system by exhibiting strategic behavior, i.e., considering other participants' objectives and stating preferences that are different from their true preferences. For example, a service seeker may consider how much delay would result if he/she seeks the resource that best matches his/her requirement as there may be several other seekers in the queue prolonging the response time. When this happens it is not optimal for a service seeker to state his/her true preferences, but rather based on expectations about other service seekers' preferences. Finally, FCFS does not account for the specific requirements of different organizational scenarios. For example, for a central service provider organization, such as Shapeways [21], where all the resources are owned by the same company and the service seekers are independent designers who are interested in printing their parts, the objective is to allocate the jobs to the resources to maximize the total utility gained by the organization. On the other hand, in a totally decentralized scenario, such as 3Dhubs [1], the utilities of all service providers and seekers need to be accounted for in the matching algorithm.

To address the limitations of the FCFS matching mechanism, the central question addressed in this paper is: *How can service seekers be optimally matched to service providers in different decentralized design and manufacturing scenarios, considering the true preferences of all agents?* We propose the use of matching theory, which has been used for different matching problems such as matching students to schools, kidney donors to patients for transplant, and residents to hospitals. To the best of our knowledge, this is the first application of matching theory within the CBDM context. We analyze the applicability of different matching algorithms in different decentralized design and manufacturing scenarios. The effects of strategic behavior of participants on the efficiency of the matching are analyzed. We also study the influence of dynamic entry and exit of agents on the optimality of matches, which is crucial in a CBDM framework. Finally, we draw insights on the effects of market thickness and resource availability on these matching algorithms through simulation studies.

The paper is structured as follows. In Section 2, we discuss a specific problem of matching designers with 3D printing service providers, along with different organizational scenarios. The steps in matching seekers and providers, and three specific algorithms used in the paper are discussed in Section 3. The three algorithms are evaluated for different scenarios in Section 4. Simulation results are presented in Section 5, and closing comments in Section 6.

2. Matching designers and manufacturers – a specific problem in CBDM

2.1. Illustrative example: 3D printing services

Additive manufacturing is bridging the gap between designers and manufacturers by enabling rapid transition of concepts into physical prototypes and final products. The increasing popularity of additive manufacturing is partly due to the availability of mid-level consumer grade 3D printers, and access to robust 3D modeling software for the creation of geometric models.

To serve designers for whom it is economically not viable to own different printers for their needs, there has been an emergence of service organizations, such as Shapeways [21], who own a variety of 3D printing machines. The machines range from desktop printers for plastic parts to industrial scale metal printers, giving designers a myriad of options to choose from based on their needs. Designers can submit geometric models to these organizations and get them printed at the quoted price. These organizations typically also offer quality checks and assistance to designers to help them market and sell their products in return for a commission.

In addition to 3D printing service organizations, an alternate, decentralized scenario exists where designers who do not possess the necessary prototyping resources are able to connect with independent individuals who own those resources. These interactions are facilitated by service matching organizations, such as 3D Hubs

[1], where designers upload their 3D models and are able to choose from the available providers. Machine owners can register their services at these platforms and advertise their printing resources, and designers can avail these services on a FCFS basis by choosing the machines that best suit their needs. The machine owner completes the 3D printing task for a price decided based on designer's requirements.

The scenario is illustrated in Fig. 1. Each designer and machine owner has different preferences from each other. The designers' preferences for machines are based on machine-owner's attributes such as resolution, tensile strength, and maximum build size. Resolution is an important attribute because the quality, detail capability, finish and accuracy of the final printed part depend on it. The strength of the final part affects its usability as a functional prototype. The machine-owner preferences are based on design attributes, such as resolution requirement, geometric properties, and printing time. Resolution is important to machine-owners because (i) the machine-owners prefer to fully utilize their resolution capabilities to maximize profits, and (ii) machine-owners may prefer to print other products in the same run and would consider a particular resolution range as an ideal match. Geometric properties, such as size, are important because they affect needs for the machine's build area.

2.2. Desirable properties of optimal bipartite matching in CBDM

Based on their preferences, machine owners can rank order the designers; and designers can rank order the machine owners. Each designer would like to be matched to a machine owner who is at the top of their rank ordered list. Similarly, each machine owner would like to be matched to the top designer in his/her list. The match is easy if each machine owner is at the top of only one designer's rank ordered list, and each designer is at the top of only one machine owner's list. However, this is a very restricting case, and is generally not true. In real scenarios, many designers may be interested in using the same 3D printer. Therefore, it is rarely possible for everyone to achieve their first preference. Given that some of the participants will not achieve their first preference, the goal is to match designers and machine owners in a way that is optimal, in some sense, for the market as a whole.

Since there are multiple decision makers with different objectives, optimality can be defined in many ways. One possible definition is based on the maximization of total utility achieved by a set of agents (e.g., the set of all service providers). Such an optimal matching can be achieved by generalized assignment algorithms, such as the Munkres algorithm [19]. Such algorithms are appropriate only if the set of agents belongs to the same organization, and have a collective preference for the entire set. The algorithm fails when agents in the set are independent and the utilities of both the service seekers and the providers need to be considered.

In addition to optimality, the match should also have a number of other desirable properties. First, the matches should be compatible with the preference structures. For example, there should not be any designer–manufacturer pair, who prefers to be matched with each other, but are not matched by the CBDM system. If there is such a pair, then that pair has an incentive to collude outside the CBDM platform. Second, participants can dynamically enter or exit the system in a decentralized environment such as CBDM. Therefore, the matching mechanism should be insensitive to participants entering and leaving the system. For example, if a matched designer–manufacturer pair leaves the system, there should not be any change in the matches for the rest of the participants. Third, addition of service providers should only help the service seekers. Finally, the system should prevent "gaming", i.e., misrepresentation of information by individuals (or group of) participants. A matching mechanism should avoid strategic behavior, which results in the stated preferences being different from the true preferences. The matching mechanism should enable participants to base their preference ordering solely on their true preferences. This is called the *truthful revelation* property of the mechanism.

These desirable properties are formally described and quantified in Section 4.1. No matching algorithm can satisfy all these properties. Based on the scenario in question, certain properties may be more important than others. For example, the likelihood of different types of gaming is different depending on the CBDM platform that the agents are operating on. Therefore, the goal is to find the best possible matching mechanism depending on the scenario. To achieve this, the first step is to analyze the possible scenarios. Three broad scenarios are discussed in Section 2.3.

2.3. Matching scenarios in CBDM

Consider three representative scenarios covering a broad range of applications: (a) fully decentralized scenario, (b) monopolistic scenario, and (c) organizational scenario. The strategic characteristics of the agents on both sides, service seekers (S) and service providers (P), are compared for the three scenarios in Table 1.

2.3.1. Fully decentralized scenario

This is a completely decentralized market scenario where independent service seekers avail services from independent service providers. The service seekers and providers will be collectively referred to as agents. The service seekers are designers who do not possess necessary resources to make physical prototypes of their designs, and service providers are companies or individuals owning resources such as CNC machines or 3D printers. It is assumed that each designer and manufacturer is an independent strategic entity aiming to maximize his/her own payoff. It is also assumed that each agent can engage in strategic behavior such as revealing preferences that are different from their true preferences.

The agents may exhibit the following types of strategic behavior: (i) a service seeker and a provider may sign contracts outside the platform, (ii) a service provider may manipulate the system by providing false information about their capacity, and (iii) service seekers may submit manipulated preference characteristics to increase the probability of matching with the desired or most sought after service providers. The objectives of the matching algorithm in the fully decentralized scenario are: it should be immune to the strategic behavior of agents, it should optimize the utility of both service seeker and provider, the optimal matching should remain the same even if some agents leave the system once matching is done.

2.3.2. Monopolistic scenario

In this scenario, a single organization (e.g., Shapeways [21]) owns a wide variety of resources and independent designers avail services from this organization. Designers submit their design requirements to these companies and get them printed or manufactured. In this scenario, the service provider is a single agent or company, which is also responsible for matching service seekers to the resources. Hence, the organization does not exhibit strategic behavior. Unlike in the totally decentralized scenario, here both the resource provider and algorithm implementer is the same agent whose sole objective is to maximize its utility by matching seekers to the resources.

The objective of an appropriate matching mechanism is only to maximize the payoff of the organization. The utility gained by the service seekers is not considered here. The preferences of the service seekers are indirectly accounted for through customer satisfaction, which is generally a part of the service-provider's preferences.

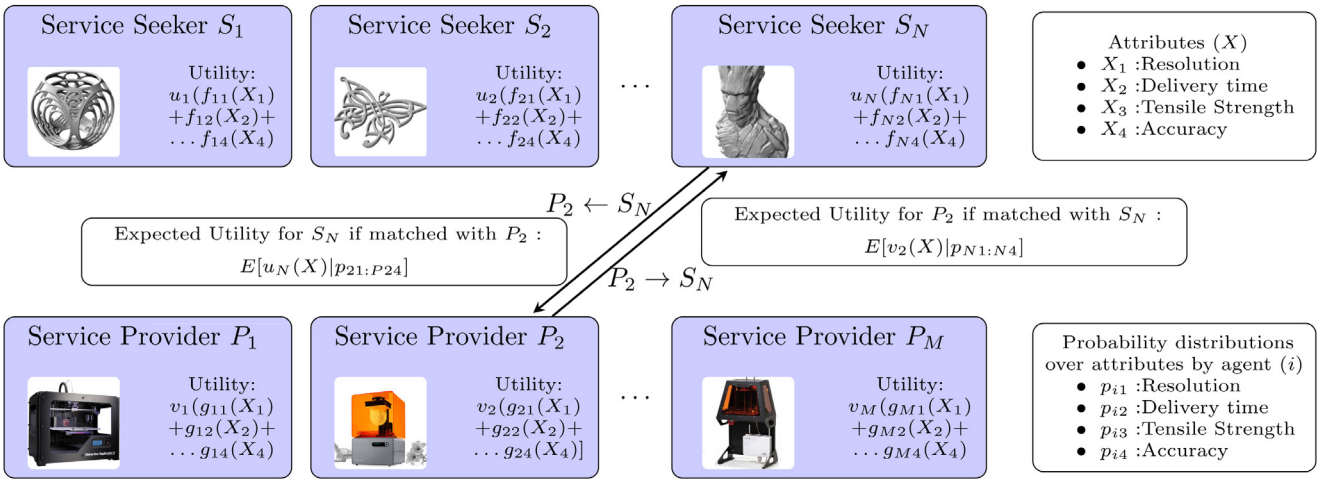

Fig. 1. Matching in decentralized design and 3D printing services.

Table 1
Comparison of strategic behavior of agents in the three scenarios.

Scenario	Set size	Service seekers (S)	Service providers (P)	Coalitions
Fully decentralized	$\|S\| > 1$, $\|P\| > 1$	Strategize	Strategize	Unlikely
Monopolistic	$\|S\| > 1$, $\|P\| = 1$	Strategize	Not Strategize	No coalition
Organizational	$\|S\| > 1$, $\|P\| = 1$	Strategize	Not Strategize	Likely

2.3.3. Organizational scenario

This scenario is different in the sense that both service seekers and service providers belong to the same organization. All the service providers are owned by the same organization and hence are non-strategic in nature. This scenario can represent students of a university availing 3D printing resources of the university to print their design projects, or designers in a R&D company printing prototypes to validate their designs.

Service seekers who belong to the same company have a greater incentive to strategize as the information and strategies about service seekers and service providers are readily available. The probability of strategic coalitions is higher in this scenario as the agents belong to the same organization. An appropriate matching mechanism should be strategy-proof to coalition formation. It should also consider the utility functions of both the service seekers and service providers.

3. Steps for matching

We propose three steps in optimal matching of designers and manufacturers within the CBDM framework. The steps are shown in Fig. 2. The first step involves quantification of preferences of designers and manufacturers using the expected utility theory [15]. The second step involves ranking of alternatives of each participant based on the expected utilities. The third step is to analyze the CBDM scenario and utilize the matching algorithm based on the most relevant criteria based on interaction, objectives and private information of agents. These steps are discussed in detail in Sections 3.1 through 3.3.

We consider a set of service seekers (i.e., designers), $S = \{s_1, s_2, \dots, s_{|S|}\}$, availing manufacturing services from a set of service providers (i.e., manufacturers), $P = \{p_1, p_2, \dots, p_{|P|}\}$. Service seekers and service providers are collectively referred to as agents $A = S \cup P$ and total number of agents $|A| = |S| + |P|$. Agents in P constitute the alternative set for agents in S, and vice-versa. Each service provider, p_j, offers a cap on the maximum number of service seekers it can serve, which we call vacancy denoted by q_{p_j} with $q_{p_j} \geq 1$. We assume that the cap is in terms of the number of designs a service provider can manufacture, and not the amount of time the manufacturer is willing to work.

3.1. Quantification of preferences of service seekers and service providers

We use the utility-based procedure for quantification of agents' preferences, as demonstrated for Fernández et al. [11]. To quantify the preference characteristics of the agents, the first step is to identify all the attributes that service seekers and service providers consider while evaluating their preferences. All service seekers in S value a set of attributes of the service providers in P, e.g., machine resolution and the attributes of the material offered. Similarly, the service providers in P value certain service-seeker attributes such as printing time. We represent the union of service seeker and service provider attributes as $X = \{X_1, X_2, \dots, X_n\}$. In the rest of the paper, s_i and p_j denote ith service seeker and jth service provider, respectively, while X_k denote the kth attribute.

Based on the preferences for each attribute X_k, service seeker s_i's single-attribute utility functions f_{ki} are obtained using standard utility assessment procedures (see [15] for details). The single attribute utility functions f_{ki} are then combined into s_i's multi-attribute utility function, u_i. We have $u_i(X) = u(f_{1i}(X_1), f_{2i}(X_2), \dots, f_{ni}(X_n))$. For illustration, assuming the additive form of the multi-attribute utility function, s_i's utility function is $u_i(X) = \sum_{k=1}^{n} w_{ki} f_{ki}(X_k)$, where w_{ki} is the weight that s_i associates to attribute X_k, such that $\sum_k w_{ki} = 1$. Note that each service seeker $s_i \in S$ may care only about a subset of relevant attributes in the set X. These subsets are generally different for different agents, particularly between service providers and service seekers. The weights for those attributes that the agents do not care about are assigned as zero.

The next step is to define the probability function of the attribute value. For service seeker s_i we are interested in the probability function of the attributes of service provider p_j. We represent the pdf of attribute X_k associated with service provider p_j as $p_{kj}(X_k)$. The probability distribution and the multi-attribute utility functions are combined to obtain the expected utility that s_i gains by being

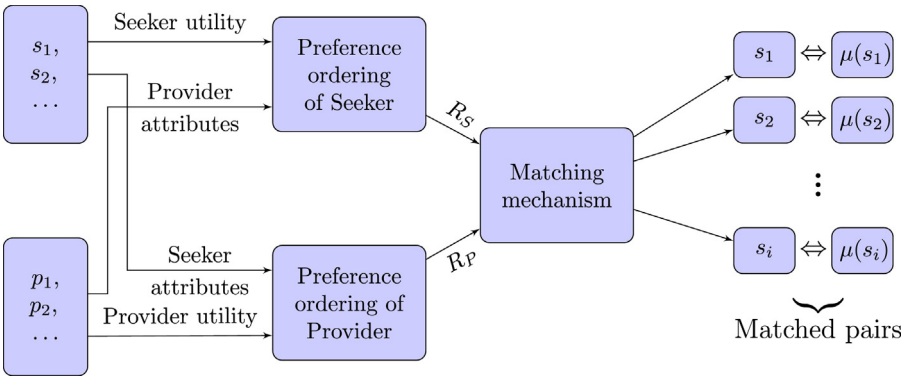

Fig. 2. The above flow chart summarizes the approach followed.

matched to each of the s_i's alternatives p_j. For the case of additive multi-attribute utility functions the following equation is valid for expected utility calculation,

$$E[u_{ij}(X)] = \sum_{k=1}^{n} w_{ki} \int [f_{ki} p_{kj}] dx \tag{1}$$

where $E[u_{ij}(X)]$ represent the utility of service seeker s_i for provider p_j. Repeating the same steps for service providers, the expected utility gained by service provider p_j being matched to each of its alternatives s_i is given by

$$E[u_{ji}(X)] = \sum_{k=1}^{n} w_{kj} \int [f_{kj} p_{ki}] dx \tag{2}$$

In cases where additive independence of attributes is not valid the expected utility expression is re-formulated. However, a detailed discussion of such special cases is beyond the scope of this paper. Fernández et al. [11] provide additional insights on such reformulations for the 3D printing scenario.

3.2. Ranking of alternatives

Based on the expected utilities thus generated, the alternatives for each agent are rank ordered. Higher the expected utility, lower is the rank. We include the agent itself in the set of its alternatives. All the alternatives that are unacceptable to the agent are ranked below the agent. An agent matched to itself represents an unmatched agent. For example, if the design cannot be printed on any of the available machines because the volume exceeds the capacity of each machine, then the design remains unmatched to any of the machines. Therefore, the set of alternatives for a seeker (s_i) is the set of providers ($P \cup s_i$), and the set of alternatives for a provider (p_j) is the set of seekers ($S \cup p_j$).

The ranking of the alternatives of each agent s_i is represented as R_{s_i}. The preference ordering thus generated is assumed to be complete, anti-symmetric and transitive as utility theory is based on assumption of rational behavior. We define $R_{s_{-i}}$ as the preferences of all service seekers except s_i and R_S as the preference ordering of all service seekers combined. We have $R_S = R_{s_{-i}} \otimes R_{s_i}$. Here, $\otimes$ symbolizes the product of vector spaces. The preference ordering of each agent is a vector and falls in a vector space. Preference ordering of a collection of agents falls in product of these vector space. Similarly, for all service providers, $R_P = R_{p_{-j}} \otimes R_{p_j}$.

3.3. Matching algorithms

Matching algorithms are aimed at implementing matching mechanisms [16]. In this paper, we use the words *mechanism* and *algorithm* interchangeably. A mechanism is a mathematical structure that models institutions through which economic activity is guided and coordinated [13]. A mechanism results in a matching $\mu : S \to P \cup S$ that

(i) assigns each service seeker to an alternative, i.e., $\forall s_i \in S : \mu(s_i) \in P \cup s_i$, where $\mu(s_i)$ is the service provider to which service seekers s_i is matched and
(ii) implements the matching without exceeding the vacancy of any service provider, i.e., $\forall p_i \in P : |\mu^{-1}(p_i)| \leq q_{p_i}$; where $\mu^{-1}(p_i)$ is the set of service seekers matched to service provider p_i

The best matching mechanism μ^* matches S and P with the preference profile $R = R_S \otimes R_P$ in the most optimal way. The notion of optimality is based on the scenario and the desired properties for that scenario.

In the following, we present three different matching mechanisms: Deferred Acceptance (DA), Top Trading Cycles (TTC), and Munkres' mechanism.

3.3.1. Deferred acceptance (DA) mechanism

The first DA mechanism was proposed by Gale and Shapley [12] as a solution to the stable marriage problem.[1] Gusfield [7] discusses the extensions and implications of this mechanism in detail. This mechanism guarantees a stable matching solution in finite time. Even 50 years after it was first formulated, DA is still applied in the National Residency Matching Program [2] where medical residents are assigned to hospitals based on the preference ordering of both residents and hospitals. It has also been used in matching workers to firms [6]. Within the service seeker and service provider context, the mechanism can be used as follows:

Algorithm 1. Deferred Acceptance (DA) mechanism

Set each $s_i \in S$ as unassigned and each $p_i \in P$ as totally unsubscribed
while ($\exists p_i \in P$ who is undersubscribed) and ($\exists s_k \in R_{p_i}$ not provisionally assigned to p_i) **do**
1. s_i is first such s_k in R_{p_i} and s_i is provisionally assigned to p_j
2. unassign s_i from p_j and provisionally assign s_i to p_i
3. for each successor p_k on R_{s_i} remove p_k and s_i from each other's list
end while

3.3.2. Top trading cycles (TTC) mechanism

TTC was proposed by Shapley and Scarf [22]. TTC has been successfully applied for kidney exchange [20] and matching students to schools [4]. The matching generated by TTC enjoys useful

[1] Extensions of the basic mechanism have been proposed, for example [18]. Here we focus only on the fundamental mechanism.

properties such as group-strategy proof and efficiency, but lacks stability (see further details in Section 4.2). The mechanism is implemented in the service matching scenario as follows:

Algorithm 2. TTC mechanism

Set each $s_i \in S$ as unassigned and each $p_i \in P$ as totally unsubscribed
while $(\exists p_i \in P)$ or $(\exists s_i \in S)$ **do**
1. All $s_i \in S$ points to top preferred $p_j \in P$ and vice-versa;
2. any agent who prefers to remain unmatched points to itself forming self-cycle;
3. each s_i is allotted the p_j it points to and vice-versa;
4. all allotted s_i are removed and capacity of all allotted p_j is reduced by one;
5. remove all p_j whose capacity reduces to zero
end while

3.3.3. Munkres' mechanism

Munkres' mechanism [19] can optimize the expected utility only for one set of agents. Unlike TTC and NRMP, Munkres optimizes the absolute value of expected utility attained by one set of agents. The details of the mechanism are as follows:

Algorithm 3. Munkres' mechanism

Populate matrix $B_{o\times o}$, where $o = \max(|S|, |P|)$ such that element $b_{i,j}$ of B represent utility of $s_i \in S$ for $p_j \in P$; empty cells in the matrix are replaced by the largest entry.
while the number of lines covering the zero elements equals the number of rows in matrix B **do**
1. For each element in a row subtract the minimum value;
2. repeat the above step for columns;
3. cover each zero in the matrix with minimum lines
4. add the minimum uncovered element to every covered element; if covered twice, add the minimum element twice
end while
while either $S \neq \phi$ and $P \neq \phi$ **do**
1. select a matching pair by choosing a set of zeros
2. reduce the capacity of matched manufacturers by one;
3. remove those $p_j \in P$ whose capacity reduced to zero and remove those $s_i \in S$ who were matched;
4. repopulate matrix B with the updated S and P; **end while**

4. Evaluation of matching mechanisms for CBDM

4.1. Criteria for evaluation

This section describes the set of criteria relevant to a diverse set of matching scenarios in CBDM framework. The important criteria include (i) individual rationality [5], (ii) stability [12], (iii) consistency [10], (iv) resource monotonicity [8], (v) population monotonicity [9], (vi) strategy proofness [16], (vii) group strategy proofness [16], (viii) Pareto efficiency [5], (ix) absolute majority [3], and (x) effective cardinal efficiency [29].

4.1.1. Individual rationality

A mechanism μ is individually rational if (i) every agent matched through μ prefers its matched partner over being unmatched, and (ii) there is no under-utilized service provider who is more preferred by any service seeker than its match. Here, an under-utilized service provider p_j is one who is matched to a lower number of service seekers than its capacity q_{p_j}. Mathematically, individual rationality can be written as:

(i) $\forall s_i \in S$, $\mu(s_i) \succeq_{s_i} s_i$,
(ii) $\nexists s_i, p_j$ such that $p_j \succ_{s_i} \mu(s_i)$ and $q_{p_j} > \mu^{-1}(p_j)$.

Practical examples where individual rationality gets violated are: (a) a company, outsourcing a manufacturing job to the cloud to save time and cost incurred to the company, matched to a service provider who demands a higher price than carrying out the job in-house; (b) a service seeker's requirements exceed the capacities of a service provider being matched to (e.g., design volume exceeding the build volume of the 3D-printer owned by the service provider). In all these examples the agent is better off remaining unmatched. Hence, the matching mechanism should eliminate those match combinations where individual rationality assumption breaks.

4.1.2. Stability

The mechanism is said to be pairwise stable if it is individually rational and has no blocking pair. Service seeker s_i and service provider p_j are said to be a blocking pair in μ if

(i) a different service seeker s_k is assigned to p_j under the mechanism μ, i.e., $\mu(s_k) = p_j$,
(ii) s_i strictly prefers p_j over its assignment, i.e., $p_j \succ_{s_i} \mu(s_i)$, and
(iii) reciprocally, p_j strictly prefers s_i over its current assignment s_k, i.e., $s_i \succ_{p_j} s_k$.

If a mechanism is not stable then there is an incentive for the blocking designer-manufacturer pair to collude outside the CBDM application platform affecting the efficiency of the matching process.

4.1.3. Consistency

The mechanism is consistent if the optimal allocation remains the same even if some agents leave along with their matched pairs. For a consistent mechanism μ, if

(i) $\phi \subset S' \subseteq S$ and $\phi \subset P' \subseteq P$, and
(ii) $\mu : S \to S \cup P$ and $\mu' : S' \to P' \cup S'$

then $\mu(s_i) = \mu(s'_i)$ for all $s_i \in S$ and $s'_i \in S'$; where S' and P' are the set of designers and manufacturers who leave the matching mechanism after matching is performed. CBDM is a dynamic system with large number of agents entering and leaving the system to seek and provide services. In such a setting, it cannot be guaranteed that the agents would accept the assignment generated by the platform. Consistency property ensures that the efficiency of a matching mechanism is not lost due to dynamic entry and exit of agents.

4.1.4. Monotonicity

The mechanism is called monotone in a bilateral matching if the welfare of each agent on one side increases (decreases) by addition (removal) of agents from the other side. If agents on the service provider side are the ones being added or removed and the welfare achieved by the each service seeker either strictly increases or decreases, then we call the matching mechanism resource monotone. Mathematically, if P' is a set of service providers different[2] from set P, with either $P' \subseteq P$ (some service providers left from original set P) or $P \subseteq P'$ (new service providers joined the original set P), then either $\mu_{R_S \otimes R_P}(s_i) \succeq \mu_{R_S \otimes R_{P'}}$ (welfare of all service seekers decrease from P to P') or $\mu_{R_S \otimes R_P}(s_i) \preceq \mu_{R_S \otimes R_{P'}}(s_i)$ (welfare increase from P to P') for each $s_i \in S$. Similarly the mechanism is called population monotone if the welfare of each resource provider is increased (decreased) by the addition (removal) of service seekers.

For example, consider matching in '3DHubs' [1]. Addition of a new machine into the mechanism should only help the designers in getting a higher ranked match, and addition of a new designer should only increase the number of suitable matches available to the service provider. Monotonicity ensures that this property holds true for matching. Monotonicity also provides the added

[2] It can also be the same set of service providers offering a reduced or increased cap on vacancy i.e., $q_{p_i} \neq q^{p_i}$.

advantage that welfare of each service seeker increases by adding a new machine, and not just the total welfare of all service seekers.

4.1.5. Strategy-proof

Mechanism μ is strategy-proof if no single agent is better off misrepresenting the preferences. Agent s_i would exhibit strategic behavior if $\mu_{R_{s_{-i}} \otimes R'_{s_i}}(s_i) \succ_{s_i} \mu_{R_{s_{-i}} \otimes R_{s_i}}(s_i)$, where R is the real preference structure and R' is the misrepresented preference structure. If the rule is immune to such behavior then it is strategy-proof. Consider the case where agents repeatedly outsource their needs to the CBDM framework. Over time the agents may learn how the matching takes place and the matching mechanism is susceptible to loss in efficiency due to strategic behavior from the agents. For example, in FCFS a designer can increase the probability of being matched to a highly sought after machine owner by switching the first and second choice. The machine owners could misrepresent the capacity of machines to increase the probability of getting matched. Hence, there is a need to make the platform immune to such strategies.

4.1.6. Group strategy-proof

The mechanism is group strategy-proof if even a coalition of agents is not better off misrepresenting the preference ordering of all the individuals in the coalition. Agents in a set $S'(\subseteq S)$ have an incentive to form coalition and falsify their preference ordering in a rule μ if (i) all agents in set S' do not decrease their welfare by colluding, and (ii) at least one agent strictly increases its welfare.

(i) $\mu_{R_{S/S'} \otimes R'_{S'}}(s_i) \succeq_{s_i} \mu_{R_S}(s_i) \forall s_i \in S'$ and
(ii) $\mu_{R_{S/S'} \otimes R'_{S'}}(s_i) \succ_{s_i} \mu_{R_S}(s_i)$ forsome $s_j \in S'$.

If agents who participate in the matching know each other well then the mechanism would be susceptible to coalition strategies [7]. In the scenarios discussed this gaming behavior is probable in the organizational scenario where the agents involved in the matching process are co-workers.

4.1.7. Pareto-efficiency

A mechanism μ is Pareto-efficient with respect to a set of agents if there is no other mechanism μ' that strictly increases the utility of a subset of agents keeping the utility of the rest of the agents the same. Rule μ is Pareto efficient with respect to set S if $\forall R \nexists \mu'$ such that

(i) $\mu'(s_i) R_{s_i} \mu_{s_i}$; $\forall s_i \in S$, and
(ii) $\exists s_k \in S$ with $\mu'(s_k) P_{s_k} \mu_{s_k}$.

Similarly, the definition can be extended to Pareto efficiency with agents of set P. Mechanism μ is Pareto efficient if it is efficient with respect to both S and P.

4.1.8. Absolute majority

A mechanism is said to satisfy the absolute majority property if it maximizes the number of agents who are matched to their first choice in their submitted preference ordering. In many cases, the designers would rate their first desired manufacturer much above the subsequent ones; they would be indifferent between second and third choices. Hence, it is always desirable to have a mechanism that matches the maximum number of agents to their first choice.

4.1.9. Effective cardinal efficiency

For sets S and P, effective cardinal efficiency is the total expected utility achieved by each agent in sets S and M, respectively.

Table 2
Comparison of mechanisms in terms of its properties.

Criterion	DA	TTC	Munkers
Individual rationality	√	√	√
Stable	√	If strongly acyclic	×
Resource monotone	If weakly acyclic	If strongly acyclic	×
Population monotonic	If weakly acyclic	If strongly acyclic	×
Consistency	If weakly acyclic	If strongly acyclic	×
Strategy-proof	√	√	×
Group-strategy proof	If weakly acyclic	√	×
Pareto efficiency	If weakly acyclic	√	√
Absolute majority	×	×	×
Effective cardinal efficiency	×	×	√

4.2. Using the criteria to compare the mechanisms for the CBDM problem

Table 2 shows competing properties while implementing mechanism design for decentralized design and manufacturing in a strategic setting. There is no universal mechanism that satisfies all these properties. For example, no mechanism is both stable and efficient [5]. However, depending on the scenario, the mechanism that has the properties most relevant to the scenario in concern can be chosen.

DA mechanism can be implemented in two ways, based on the set of agents that proposes [7]. DA mechanism has the property that it is optimal with respect to the agents who propose during its implementation. Thus, if the service providers in P propose, then it is P-optimal, whereas if the seekers propose then it is S-optimal. The former provides the optimal stable matching for agents of set P whereas the latter results in optimal matches for agents in set S. In the rest of the paper, these mechanisms are labeled PODA (Provider Optimal DA) and SODA (Seeker Optimal DA), respectively. In addition to being P-optimal, PODA is strategy-proof with respect to agents in P however agents in S can strategize. Similarly, SODA is strategy-proof only with respect to agents in S. Thus PODA is not group-strategy proof with agents in S. Kojima [17] shows that PODA is not group-strategy proof if there is at least one $p_i \in P$ with $q_{p_i} > 1$. In such cases one can generate scenarios where agents in P can manipulate via misreporting vacancies or being involved in prearranged matches outside the application platform.

DA always produces stable matching. It is resource monotonic and population monotonic but not always consistent. Although DA Pareto dominates any other stable allocation, it is not optimal. Also, DA is strategy proof but not group-strategy proof. Whereas TTC is efficient and group-strategy proof but lacks consistency and is not resource and population monotone.

Unlike TTC and DA which only account for the ordinal ranking of the preference ordering, Munkres mechanism maximizes the cardinal value of total expected utility for a set. Hence, when the aim is to maximize expected utility of non-strategic agents belonging to the same set (either S or P) then Munkres is the best choice. For example, organizations such as Shapeways can adopt Munkres mechanism to match the uploaded designs to the right machines. But it breaks down in 3Dhubs where the individuals agents in either set are utility maximizing strategic agents. Moreover, Munkres mechanism is not population monotone, resource monotone, consistent, stable and strategy proof. In a bipartite matching situation the utility of the two sets of agents needs to be maximized. Munkres can optimize only a single set at a time.

5. Simulation results

In the decentralized service seeker-provider scenario, discussed in Section 3, there are *Permutation* $\left(\sum_i^y q_{p_i}, x\right)$ possible unique matching combinations. The best mechanism for a scenario is the

one that best satisfies the properties in Table 2. In Section 4, a comparison of the performance of the mechanism is presented based on various properties. The comparison does not provide information about the effects of uncertainty, such as uncertainty in the order of arrival of service requests and preference characteristics. Additionally, the implications of scarcity or abundancy of resources on the performance of the mechanisms are also not clear based on these properties. As an example, if resources are abundant such that for every new service request there is an available slot on the first preferred resource provider then FCFS would perform well. To analyze the effects of uncertainty and resource availability, simulation studies are carried out. Simulation results related to the performance of the mechanisms under scarce, balanced and surplus supply of resource are discussed in this section.

Two measures are used to compare the efficiency of different mechanisms: (i) average rank, and (ii) total expected utility. An agent s_i has rank r if matched to its rth choice in the agent's preference ordering. The average rank of designers (manufacturers) is obtained by averaging the rank of all designers (manufacturers) matched using the mechanism. The total expected utility of designers (manufacturers) is obtained by aggregating the utility gained by all the designers (manufacturers) through the matching mechanism. By definition utility attained by an unmatched designer is zero. The total utility of a set quantifies the collective performance of the set of agents, whereas average rank over all realized matches in a set quantifies the extent to which individual preferences are met. Thus, both these measures are used to evaluate the performance of different matching mechanisms.

The following three cases of resource availability are simulated:

(i) resource scarce case, where $\sum_i^y q_{p_i} < x$;
(ii) resource balanced case, where $\sum_i^y q_{p_i} = x$; and
(iii) resource surplus case, where $\sum_i^y q_{p_i} > x$.

As an illustrative scenario, there are 25 designers being matched to 5 manufacturers. Assuming that all manufacturers serve the same number of designers, $q_{p_i} < 5$ represents a resource scarce case, $q_{p_i} = 5$ is a resource balanced case, and $q_{p_i} > 5$ is resource surplus. To represent the three cases, simulation results are presented for $q_{p_i} = 3$, $q_{p_i} = 5$, and $q_{p_i} = 7$ for each manufacturer. The insights drawn are consistent for other values of q_{p_i} as well.

5.1. Illustrative example: 3D printing service framework

Consider a scenario where a set of 25 designers ($|S| = 25$ where service seeker set S is the set of designers) are seeking services from a set of 5 manufacturers ($|P| = 5$). The 25 designs used in this example were downloaded from Thingiverse [28]. Each of the 5 manufacturers owned one of these five 3D printers: Makerbot 2 (P1), Ultimaker 2 (P2), Witbox (P3), B9 Creator (P4), Form 1+ (P5). First three machines are FDM (Fusion Deposition Material) machines while the last two are based on the SLA (StereoLithography) process. Each manufacturer owned materials compatible with their respective machine. Material data was collected from iMaterialise [14] for different FDM and SLA materials. Representative images of the designs used are shown in Fig. 3.

The manufacturer attributes concerning the designers are machine volume, machine resolution, material tensile strength, manufacturer proximity whereas designer attributes concerning the manufacturers are printing time, material requirement, and design dimensions. The resolution capabilities of the machines varied from 5 to 100 μm. Designers could list their job on an urgency scale from 1 to 5. Further the designers could state their resolution and tensile strength range requirements. The numerical values used for the design printing time and machine dimension attributes are listed in Table 3.

Table 3
Printing time and design dimensions corresponding to each designer-manufacturer pair. The machines are numbered from p_1 to p_5 and 25 designs are numbered from s_1 to s_{25}, The 25 designs used in this example were downloaded from website thingiverse [28]. 3D printers used were Makerbot 2 (p_1), Ultimaker 2 (p_2), Witbox (p_3), B9 Creator (p_4), Form 1+ (p_5). Material data was collected from iMaterialise [14]. Empty cells denote incompatible design-machine pairs.

	Printing Time in machine type (h)					Area (cm^2)	Vol (cm^3)
	p_1	p_2	p_3	p_4	p_5		
s_1	0.28	0.35	0.40	0.12	0.17	18.6	15.55
s_2	0.15	0.13	0.15	0.13	0.15	304.41	94.08
s_3	0.13	0.08	0.12	0.25	0.20	407.77	13.44
s_4	0.03	0.03	0.05	0.33	0.23	0.62	0.04
s_5	2.27	1.77	1.83	0.72	0.70	4.28	11.13
s_6	0.75	0.83	1.13	0.97	0.60	23.31	1.06
s_7	0.77	1.37	1.07	1.20	0.88	49	28.92
s_8	1.93	2.02	2.28	2.15	1.43	13.14	0.67
s_9	3.15	3.93	4.90	2.67	1.73	57.5	9.99
s_{10}	6.25	7.67	8.97	3.30	2.40	103.58	26.71
s_{11}	2.25	1.55	1.88	3.68	2.25	74.01	21.23
s_{12}	4.73	6.15	6.85	4.27	2.68	38.36	29.62
s_{13}	9.48	11.28	13.00	4.65	6.25	56.91	199.65
s_{14}	7.73	10.13	11.15	5.13	5.33	1.32	0.42
s_{15}	2.07	1.92	2.00	5.77		56.5	330.65
s_{16}	12.92	16.88	18.52	7.35	8.20	6.25	11.27
s_{17}		2.00				116.91	16.43
s_{18}	3.88	4.10	4.78		0.97	76.42	298.12
s_{19}	3.48	3.62	3.93		1.13	57.3	286.37
s_{20}		11.02	12.50			58.87	55.05
s_{21}	13.82	17.28	19.10		6.65	72.4	77.79
s_{22}	24.87		34.15			43.37	2.3
s_{23}	11.98	12.80	16.92		5.38	99.97	1227.76
s_{24}			10.37			30.98	12.73
s_{25}			48.92			340.62	54.06

Utility functions of individual agents are derived and preference ordering is generated using the procedure described by Fernández et al. [11]. An example of expected utility of service seeker (designer) s_1 for service provider (manufacturer) p_2 is described. Service provider s_1 preferred the attribute tensile strength with a weight of 0.8 and resolution with a weight of 0.2. The left hand side utility function of designer s_1 for attribute resolution x_1 is defined as $f_{11}(x_1) = -0.003\ {x_1}^2 + 0.049\ x_1 - 0.9446$ in the range $22\ \mu m < x_1 \leq 60\ \mu m$, and the right hand side utility defined between $60\ \mu m < x_1 \leq 90\ \mu m$ is $f_{11}(x_1) = -0.0004\ {x_1}^2 + 0.0333\ x_1 + 0.6$. For designer s_1, the single attribute utility for tensile strength varies from 60 MPa to 75 MPa and the function is given by $f_{12}(x_2) = -0.001\ {x_2}^2 + 0.010\ x_2 + 1.200$. Assuming additive independence of the attributes, multi-attribute utility function of designer s_1 is $u(X) = 0.2f_{11}(x_1) + 0.8f_{12}(x_2)$. Manufacturer p_2 offers resolution between 50 μm and 300 μm and tensile strength between 22 MPa and 41 MPa with uniform probability. Using these function values the expected utility gained by designer s_1 being matched to manufacturer p_2 was calculated as $E[u_{12}(X)] = 0.353$. Single attribute utility functions of manufacturer p_2 were $g_{23}(x_3) = -0.000\ {x_3}^2 + 0.0024\ x_3 + 0.1975$, $g_{21}(x_1) = -0.000\ {x_1}^2 - 0.0009\ x_1 + 1.052$ and $g_{24}(x_4) = -0.0327\ {x_4}^2 + 0.4327\ x_2 - 0.2082$. Again assuming additive independence, multi-attribute utility function for manufacturer p_2 is defined as $u(X) = 0.375g_{23}(x_3) + 0.375g_{21}(x_1) + 0.25g_{24}(x_4)$. Using these functions the expected utility gained by manufacturer p_2 being matched to s_1 was calculated as $E[u_{21}(X)] = 0.071$. Similarly, expected utilities of each agent are calculated for each of their potential matches. Some of the matches are incompatible, e.g., the design may not fit in the build envelope of the machine. In such cases, the utility of matching is zero. Table 4 shows the results of expected utilities calculated for each agent for each alternative.

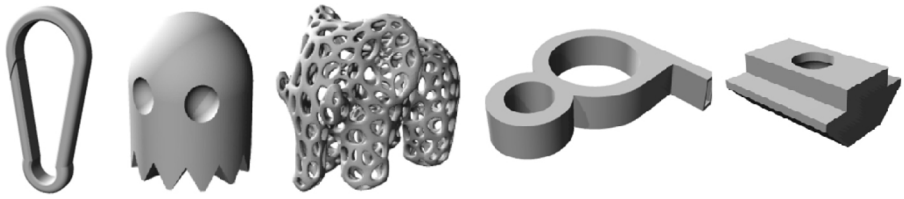

Fig. 3. Sample designs used in the illustrative scenario.

Table 4
Expected utility achieved by each agent for each of their alternatives. Numerical value tabulated in row i and column j indicates the expected utility achieved by service seeker s_i being matched to service provider p_j and the entry in bracket indicates the expected utility achieved by p_j in return. '–' represents that the match is not feasible.

Manufacturer/Designer	(p_1)	(p_2)	(p_3)	(p_4)	(p_5)
(s_1)	0.355 (0.196)	0.353 (0.071)	0.355 (0.313)	0.000 (0.191)	0.004 (0.527)
(s_2)	0.000 (0.001)	0.047 (0.001)	0.043 (0.001)	–	–
(s_3)	0.000 (0.001)	0.004 (0.002)	0.000 (0.001)	–	–
(s_4)	0.019 (0.000)	0.004 (0.001)	0.000 (0.000)	0.349 (0.088)	0.000 (0.056)
(s_5)	0.000 (0.000)	0.022 (0.001)	0.017 (0.000)	0.470 (0.088)	0.003 (0.000)
(s_6)	0.379 (0.196)	0.377 (0.071)	0.379 (0.313)	0.000 (0.197)	0.004 (0.527)
(s_7)	0.355 (0.039)	0.453 (0.021)	0.326 (0.063)	0.148 (0.11)	0.129 (0.577)
(s_8)	0.233 (0.039)	0.324 (0.021)	0.233 (0.063)	0.094 (0.108)	0.101 (0.576)
(s_9)	0.012 (0.000)	0.005 (0.142)	0.000 (0.013)	0.430 (0.091)	0.000 (0.001)
(s_{10})	0.307 (0.196)	0.283 (0.072)	0.284 (0.314)	-	0.000 (0.542)
(S11)	0.612 (0.040)	0.875 (0.021)	0.637 (0.063)	0.246 (0.191)	0.247 (0.577)
(s_{12})	0.326 (0.097)	0.513 (0.021)	0.386 (0.063)	0.131 (0.315	0.140 (0.520)
(s_{13})	0.490 (0.040)	0.680 (0.021)	0.490 (0.063)	0.197 (0.256)	0.199 (0.521)
(s_{14})	0.431 (0.039)	0.567 (0.021)	0.408 (0.063)	0.178 (0.130)	0.161 (0.520)
(s_{15})	0.016 (0.000)	0.005 (0.001)	0.000 (0.000)	0.385 (0.091)	0.000 (0.001)
(s_{16})	0.000 (0.000)	0.004 (0.001)	0.000 (0.000)	0.282 (0.088)	0.006 (0.000)
(s_{17})	–	0.053 (0.001)	–	–	0.009 (0.001)
(s_{18})	0.031 (0.072)	0.002 (0.170)	0.000 (0.020)	–	0.000 (0.016)
(s_{19})	0.008 (0.000)	0.006 (0.050)	0.000 (0.031)	–	0.000 (0.057)
(s_{20})	–	0.629 (0.021)	0.453 (0.063)	–	0.179 (0.577)
(s_{21})	0.000 (0.000)	0.004 (0.001)	0.000 (0.000)	–	0.006 (0.001)
(s_{22})	0.227 (0.196)	–	0.287 (0.313)	–	0.011 (0.542)
(s_{23})	0.355 (0.196)	0.353 (0.072)	0.355 (0.314)	–	0.004 (0.527)
(s_{24})	–	–	0.490 (0.063)	–	0.199 (0.520)
(s_{25})	–	–	0.350 (0.063)	–	–

Based on the expected utilities the designers are matched to manufacturers using the Munkres, NRMP, TTC, and FCFS mechanisms. The total expected utility gained, and the average rank of the designers and manufacturers are then calculated for each mechanism. The utility and rank are dependent on the preferences of the agents. In FCFS these may also depend on the order of arrival of the service requests. All the illustrative numerical values listed in Tables 3 and 4 are for a particular preference distribution and the order of arrival of the service requests. Thus, there is variability associated with the performance of the mechanism. To account for the variability, several combinations of preference distributions were used, and the results were analyzed statistically. Urgency, resolution and material requirement of each designer and manufacturer and the relative weightage of these attributes were randomized for each run. For each random preference distribution so generated, all the four matching mechanisms were implemented and the overall efficiency was compared.

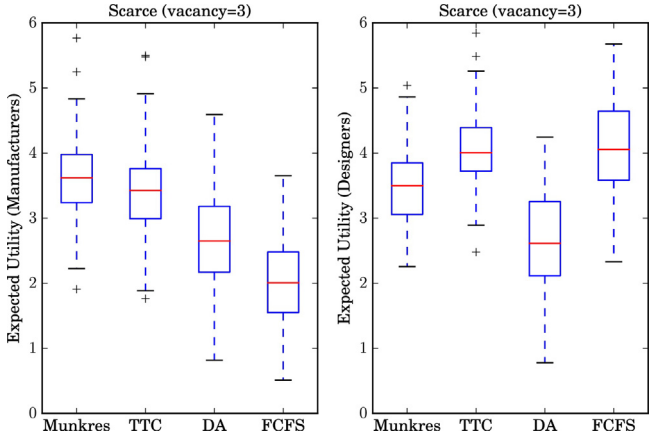

Fig. 4. Total expected utility attained by manufacturers and designers in resource scarce case.

5.2. Influence of resource availability

Figs. 4–7 show a comparison of the total utility and average rank by each of the mechanisms under various resource conditions. All the mechanisms were run for 100 randomly generated preference ordering for each of resource scarce, resource balance and resource surplus setting. The boxplots of the total utility attained by designers and manufactures under various resource conditions are shown in Figs. 4–6. While Figs. 4–6 show total utility, Fig. 7 shows the average rank attained by designers and manufacturers under resource balanced conditions.

Comparing DA and TTC in Fig. 4, it is observed that the total manufacturer utility gained by TTC is more than in DA even though a PODA (manufacturer optimal) mechanism is implemented. This is because PODA guarantees stability and can operate only in the stable space of solutions. But as the resource availability increases from scarce to balanced and abundant, the performance of TTC and DA is nearly similar in terms of the manufacturers' utilities. This is because the set of stable solutions grows rapidly as the resources get abundant and the formation of weak and strong cycles [7] decreases and efficiency of DA matches that of TTC with respect

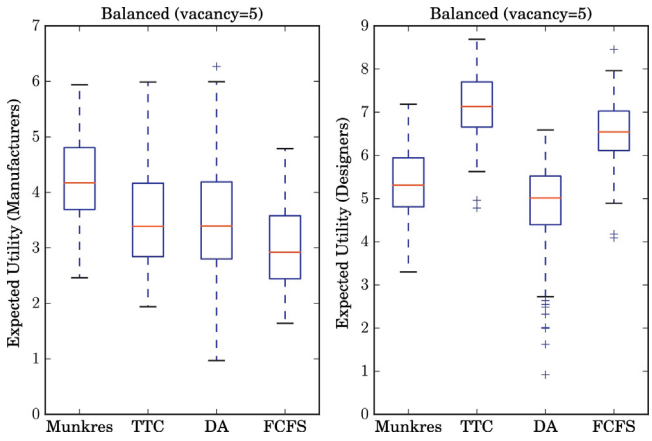

Fig. 5. Total expected utility attained by manufacturers and designers in resource balanced case.

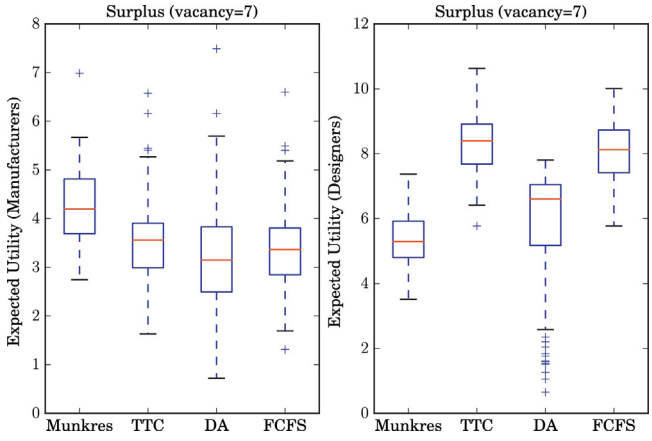

Fig. 6. Total expected utility attained by manufacturers and designers in resource surplus case.

to manufacturer utility. Thus, in fully decentralized market when demand and supply of services and resources are balanced, DA performs as well as TTC for manufacturers even though it operates in the space of stable solutions. In addition DA mechanism offers stable solutions and hence a mechanism based on DA is the best mechanism in a totally decentralized CBDM market. The total expected utility of the designers by implementing TTC or DA increases from resource scarce to resource balanced, but after that it depends on the set of feasible matchings.

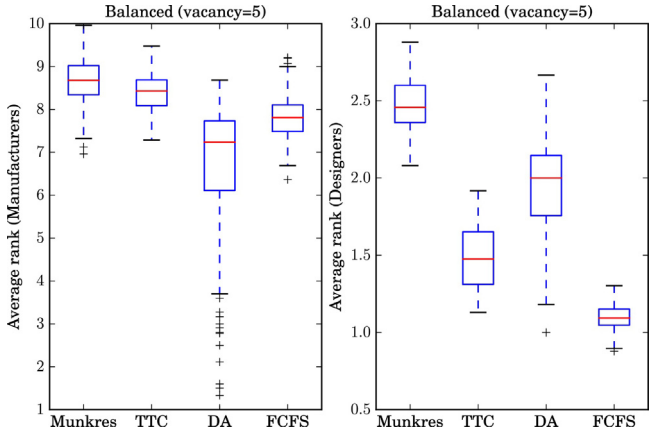

Fig. 7. Average rank over completed matches attained by manufacturers and designers in resource balanced case.

In the case of FCFS, the designer utility grows from resource scarce to resource surplus as the availability of most preferred manufacturers increases with resource availability. The average designer rank (see Fig. 7) of FCFS on the other hand is better than all other mechanisms. However, this average is only calculated over the completed matches. In FCFS the number of completed matchings is arbitrary even for a given instantiation depending on the order of arrival of service requests. Additionally, FCFS does not have any of the useful properties listed in Table 2. Therefore FCFS is not a preferred mechanism in any of the three scenarios discussed.

Munkres offers the highest manufacturer utility regardless of scarcity of resources. But the average rank attained by the manufacturers is better in DA and TTC as compared to Munkres. This is because DA and TTC are based on the ordinal preference ordering whereas Munkres is based on total cardinal utility attained by the entire set of agents. The difference between Munkres and TTC, DA is clear particularly in scarce resource settings.

5.3. Evaluation of matching mechanisms for the three scenarios

Based on critical analysis of the properties of the mechanisms and their performance under various resource conditions the best mechanisms for the three scenarios are listed in Table 5. For the *monopolistic scenario*, the best mechanism is the one based on the Munkres mechanism. From Figs. 4–6, it is observed that total expected utility attained by set of manufacturers is maximum for the Munkres mechanism, irrespective of availability of resources. In monopolistic scenario the company who owns all the resources is in charge of carrying out the matching process. Monopolist will try to maximize its utility and hence would resort to the Munkres mechanism.

In *two-sided strategic scenario* DA is the most suited mechanism in the case of totally decentralized scenario despite the lower efficiency compared to TTC. This is because it offers useful properties such as consistency, resource and population monotonicity, and stability. Balinski and Sonmez [5] show that if DA is not consistent for a problem, then TTC is not consistent for it either; the converse is not true. Similarly if DA fails to be stable, TTC also fails; but the converse is not true. Resource monotone and population monotone means even if some service providers or service seekers (or both) leave after the final matching allotments are declared, the solution for the reduced set of agents remains the same. This is highly possible in totally decentralized scenario with independent strategic agent being the stakeholders. Consistency of the DA mechanism ensures that the most optimal matching remains unchanged even if some agents leave the system. Resource and population monotone nature of DA ensures fairness even if some service seekers or providers leave. Finally, there is evidence from real world applications [2] which shows that stable mechanisms often succeed over unstable ones. Balinski and Sonmez [5] highlight this by stating that if one needs to consider both stable and optimal solutions then DA ranks ahead of TTC.

Now that we have rated DA over TTC in this scenario, still we need to choose among the two types of DA (i.e., SODA and PODA). SODA is strategy-proof with respect to agents in S, i.e., the designers, while PODA is strategy proof with respect to manufacturers only. Manufacturers have a higher probability to behave strategically because unlike designers, manufacturers can strategize both by capacity manipulation and outside settlement. Additionally, manufacturers are repeatedly involved and have more learning effect than designers. Hence we propose the use of PODA over SODA to make it strategy-proof with respect to the manufacturers. Further, we rule out FCFS and Munkres in this scenario as it takes care of only one set of agents and do not offer properties such as stability and consistency, which are important in a dynamic environment.

Table 5
Relevant properties and best mechanisms for the three scenarios.

Scenario	Relevant properties	Best mechanism
Totally decentralized	Strategy proof, stable, Resource monotonic, Population monotonic, consistent, individually rational, bilateral utility	PODA
Monopolistic	Efficiency, cardinally optimal	Munkres
Organizational	Group-strategy proof, bilateral utility, efficiency	Top-Trading Cycle

In an *organizational scenario*, we need to enforce a canonical response[3] unlike totally decentralized scenario. For example, students using 3D-printing resources in a university can group-strategize to gain unfair advantage. Therefore, in addition to optimizing the preferences of both sets of agents, the mechanism needs to be group-strategy proof as the system is more prone to coalition strategies. Therefore, the mechanism based on TTC is best suited in this scenario.

6. Closing comments

There is no single best matching mechanism for all scenarios. Depending on the strategic behavior and resource availability the right mechanism should be implemented. If the objective is to maximize the utility of a single monopolistic resource provider then Munkres is the best mechanism. When the preference ordering of both service seekers and providers need to be considered TTC and DA perform better. In a totally decentralized scenario, where stable solutions are important, DA is the best mechanism. In an organizational scenario such as students using 3D printing resources in a university, TTC is the best mechanism. Table 5 summarizes the general approach and mechanism for the three scenarios considered in this paper. The performance of the mechanisms also depends on the availability of resources which in turn is based on the market thickness. Hence, the insights drawn from the simulations can also be used to choose appropriate mechanism based on the frequency of matching and the type of the scenario.

There are several mechanism design related issues specific to CBDM for further investigation. First, the present analysis is based on the assumption that agents are substitutes and not complements. But CBDM is about integrating collective resources to get the manufacturing task done. Thus, manufacturing resources may become complements depending on the needs of the service seeker. Second, resource discovery is a challenging task in CBDM. It may be impossible for all agents to provide an exhaustive list of their alternatives. The agents are better off revealing the preference characteristics towards attributes. There is a need to design strategy proof mechanisms when agents reveal attribute preferences instead of alternatives. This is because many properties of the mechanisms change when preferences are revealed as attributes and not as alternatives. Third, CBDM involves exchange of transferable utility such as money. This assumption has been relaxed in the analysis. Fourth, the resources may not be perfectly divisible. For example, in the NRMP application, when medical residents are matched to hospitals there is a fixed vacancy to be filled. But in CBDM a few resource owners may be willing to manufacture the product for pre-determined amount of time and for the available time the resource can be divided among multiple service seekers. Thus, the matching problem in CBDM is unique as compared to the previous applications of the matching mechanisms. These differences bring in new challenges. As an example, when resources are not perfect substitutes there will not exist any stable matching.

[3] The response depends only on the individual's preference characteristics and rules of the mechanism and is not influenced by strategies and preferences of other agents.

Acknowledgements

The authors gratefully acknowledge financial support from the US National Science Foundation (NSF) through grants 1265622, 1360361 and 1329979.

References

[1] 3Dhubs. 3Dhubs kernel description; 2015 https://www.3dhubs.com/ [accessed: 11.05.15].
[2] Abdulkadiroğlu A, Pathak PA, Roth AE. The New York city high school match. Am Econ Rev 2005;95(2):364–7, http://dx.doi.org/10.1257/000282805774670167.
[3] Abdulkadiroglu A, Sonmez T. Random serial dictatorship and the core from random endowments in house allocation problems. Econometrica 1998;66(3):689, http://dx.doi.org/10.2307/2998580.
[4] Abdulkadiroğlu A, Sönmez T. School choice: a mechanism design approach. Am Econ Rev 2003;93(3):729–47, http://dx.doi.org/10.1257/000282803322157061.
[5] Balinski M, Sönmez T. A tale of two mechanisms: student placement. J Econ Theory 1999;84(1):73–94, http://dx.doi.org/10.1006/jeth.1998.2469.
[6] Crawford VP, Knoer EM. Job matching with heterogeneous firms and workers. Econometrica 1981;49(2):437, http://dx.doi.org/10.2307/1913320.
[7] Dan Gusfield RWI. The stable marriage problem: structure and algorithms. Concurr Eng 1989.
[8] Ehlers L, Klaus B. Resource-monotonicity for house allocation problems. Int J Games Theory 2004;32(4), http://dx.doi.org/10.1007/s001820400177.
[9] Ehlers L, Klaus B, Pápai S. Strategy-proofness and population-monotonicity for house allocation problems. J Math Econ 2002;38(3):329–39, http://dx.doi.org/10.1016/S0304-4068(02)00059-9.
[10] Ergin HI. Consistency in house allocation problems. J Math Econ 1999;34:77–97.
[11] Fernandez MG, Seepersad CC, Rosen DW, Allen JK, Mistree F. Decision support in concurrent engineering – the utility-based selection decision support problem. Concurr Eng 2005;13(1):13–27, http://dx.doi.org/10.1177/1063293X05050912.
[12] Gale D, Shapley LS. College admissions and the stability of marriage. Am Math Mon 1962;69(1):9, http://dx.doi.org/10.2307/2312726.
[13] Hurwicz L, Reiter S. Designing economic mechanisms. Cambridge University Press (CUP); 2006, http://dx.doi.org/10.1017/CBO9780511754258.
[14] iMaterialise. imaterialise; 2015 http://i.materialise.com [accessed: 27.03.16].
[15] Keeney RL, Raiffa H. Decisions with multiple objectives: preferences and value tradeoffs. Cambridge University Press; 1993.
[16] Kesten O. On two competing mechanisms for priority-based allocation problems. J Econ Theory 2006;127(1):155–71, http://dx.doi.org/10.1016/j.jet.2004.11.001.
[17] Kojima F. When can manipulations be avoided in two-sided matching markets? Maximal domain results. BE J Theor Econ jan 2007;7(1), http://dx.doi.org/10.2202/1935-1704.1405.
[18] McVitie DG, Wilson LB. The stable marriage problem. Commun ACM 1971;14(7):486–90, http://dx.doi.org/10.1145/362619.362631.
[19] Munkres J. Algorithms for the assignment and transportation problems. J Soc Ind Appl Math 1957;5(1):32–8, http://dx.doi.org/10.1137/0105003.
[20] Roth AE, Sonmez T, Unver MU. Kidney exchange. Q J Econ 2004;119(2):457–88, http://dx.doi.org/10.1162/0033553041382157.
[21] Shapeways. Shapeways; 2015 http://www.shapeways.com/ [accessed: 11.05.15].
[22] Shapley L, Scarf H. On cores and indivisibility. J Math Econ 1974;1(1):23–37, http://dx.doi.org/10.1016/0304-4068(74)90033-0.
[23] Tao F, Cheng Y, Xu LD, Zhang L, Li BH. CCIoT-CMfg: cloud computing and internet of things-based cloud manufacturing service system. IEEE Trans Ind Inf 2014;10(2):1435–42, http://dx.doi.org/10.1109/TII.2014.2306383.
[24] Tao F, Hu Y, Zhao D, Zhou Z. Study on resource service match and search in manufacturing grid system. Int J Adv Manuf Technol 2008;43(3–4):379–99, http://dx.doi.org/10.1007/s00170-008-1699-7.

[25] Tao F, Hu YF, Zhou ZD. Study on manufacturing grid & its resource service optimal-selection system. Int J Adv Manuf Technol 2007;37(9–10):1022–41, http://dx.doi.org/10.1007/s00170-007-1033-9.

[26] Tao F, LaiLi Y, Xu L, Zhang L. FC-PACO-RM: a parallel method for service composition optimal-selection in cloud manufacturing system. IEEE Trans Ind Inf 2013;9(4):2023–33, http://dx.doi.org/10.1109/TII.2012.2232936.

[27] Tao F, Zhang L, Nee A. A review of the application of grid technology in manufacturing. Int J Prod Res jul 2011;49(13):4119–55, http://dx.doi.org/10.1080/00207541003801234.

[28] Thingiverse. Thingiverse; 2016 http://www.thingiverse.com/ [accessed: 27.03.16].

[29] Von Neumann J, Morgenstern O. The theory of games and economic behavior. NJ: Princeton University Press; 1947.

[30] Wu D, Rosen DW, Wang L, Schaefer D. Cloud-based design and manufacturing: a new paradigm in digital manufacturing and design innovation. Comput Aided Des 2015;59:1–14, http://dx.doi.org/10.1016/j.cad.2014.07.006.

[31] Xu X. From cloud computing to cloud manufacturing. Robot Comput Integr Manuf 2012;28(1):75–86, http://dx.doi.org/10.1016/j.rcim.2011.07.002.