AOTidf Bagger Game

DENIS ERFURT, TOBIAS BEHRENS, ABDALLAH KADOUR

TODO: Abstract

---

1 GLIEDERUNG

TODO: gliederung

2 VORAUSSETZUNGEN

In diesem Kapitel geben wir in Abschnitt (2.1) einen Gesamtüberblick über die Aufgabenstellung und formulieren Prämissen für die weitere Bearbeitung. Im Abschnitt (2.2) werden weiterhin notwendige theoretische Grundlagen zusammengefasst, die für das weitere Verständnis der Arbeit notwendig sind.

2.1 Aufgabenstellung

Das gegebene Szenario lässt sich wie folgt zusammenfassen: Es existieren eine Menge an Baustellen und Baufirmen, im Folgenden Agenten genannt. Agenten verfügen über ein Bestand an Skillkapazitäten – eine Anzahl an Einheiten verschiedener Skills – und Baustellen wiederum benötigen Skillkapazitäten für ihre Fertigstellung. Wird die Baustelle fertiggestellt, wird eine vorher definierte Summe – der Erlös – ausgeschüttet.
Diese Arbeit modelliert dieses Szenario (Kapitel 3) und untersucht Mechanismen (Kapitel 4), die mit Hilfe von Koalitionsbildung eine Zuordnung von Bestand und Bedarf an Skillkapazitäten bestimmen. Die Mechanismen werden bezüglich der Maximierung der sozialen Wohlfahrt, Stabilität und Fairness der Gewinnausschüttung, sowie auch ihrer absoluten Performance untersucht.
Unter **sozialer Wohlfahrt** verstehen wir die Summe der Gewinne aller Agenten. Der Gewinn eines Agenten a ist dabei die Differenz zwischen dem Teil des Erlöses, den a aus dem Gesamterlös erhält, und den Kosten, die für a im Spiel für die Bereitstellung seiner tatsächlich Skillkapazitäten anfallen.
Ein Mechanismus ist in unserem Verständnis bezüglich seiner Gewinnausschüttung **fair**, wenn dessen Gewinnausschüttung für jeden Agenten a dem *Shapley Value* für den Agenten a entspricht. Wir folgen hiermit also dem normativen Verständnis des Shapley Values.

TODO: Def. Shapley Value hier?

Ist Gewinnausschüttung

Prämissen. Des Weiteren gehen wir von folgenden Grundannahmen bei der Bearbeitung der Aufgabenstellung aus:

(1) **Rationalität**: Agenten arbeiten ausschließlich für ihr eigenes Interesse.
(2) **Multiskill**: Agenten können mehrere Skilltypen mit beliebiger Quantität besitzen.
(3) **Linearität**: Skillkapazitäten können höchstens ein mal eingesetzt werden und sind nach ihrem Einsatz ”verbraucht”.
(4) **unvollständige Information der Konkurrenz**: Agenten haben keine Information über die Ressourcen anderer Agenten und treten gegeneinander in Konkurrenz.

(5) **vollständige Informationen des Bedarfs**: Agenten haben vollständige Information über die Anzahl, Position, Bedarf sowie Vergütung der Bauaufträge.
(6) **Zeitagnostisch**: Alle Betrachtungen werden ohne Berücksichtigung der Zeit gemacht. Insbesondere erfolgen über die Zeit keine Änderungen an den Bauaufträgen oder den Skillkapazitäten Agenten.

2.2 Grundlagen

Im weiteren Verlauf der Arbeit werden Funktionen mit kleinem Anfangsbuchstaben und Relationen mit einem großen Anfangsbuchstaben geschrieben. Ebenfalls gehen wir davon aus, dass alle Funktionen total und mit 0 initialisiert sind, falls für eine Struktur Eingabeparameter nicht näher definiert werden. Wir verwenden die Schreibweise in Prädikatenlogik und die Mengenschreibweise äquivalent: $a \in A \equiv A(a)$ Um das gegebene Problem strukturell analysieren zu können, benutzen wir die Sprache der HOL (Higher Order Logic).

3 MODELLIERUNG

In diesem Kapitel verfolgen wir das Ziel, durch eine Formalisierung der Aufgabenstellung eine geeignete Grundlage zu für die Formulierung von Mechanismen und die Beantwortung zentraler Fragestellungen zu schaffen. Zunächst werden wir in Abschnitt (3.2) grundlegende Definitionen vorstellen sowie in Abschnit (3.1) zwei Signaturen sowie die dazugehörigen Modellklassen einführen.

3.1 Signaturen und Modellklassen

Wir führen nun zwei verschiedene Signaturen mit den dazugehörigen Modellklassen ein:

(1) **CSGS** - Coalition Skill Game Setting
(2) **CSG** - Coalition Skill Game

Definition 3.1 (CSGS). Eine Coalitional Skill Game Setting-Signatur (CSGS-Signatur) sei definiert als

$$\sigma_{CSGS} :=$$
$$\{Agent_{/1}, Baustelle_{/1}, supply_{/2}, demand_{/2}, budget_{/1}, kosten_{/4}\}$$

Intuition

$Agent(x)$	$:\Leftrightarrow$	x ist ein Agent (Baufirma)
$Baustelle(x)$	$:\Leftrightarrow$	x ist eine Baustelle
$supply(x,t) \mapsto n$	$:\Leftrightarrow$	Agent x besitzt n Einheiten vom Skilltyp t
$demand(x,t) \mapsto n$	$:\Leftrightarrow$	Baustelle x benötigt n Einheiten vom Skilltyp t
$budget(x) \mapsto n$	$:\Leftrightarrow$	Baustelle x ist bereit einen Gewinn von maximal n bei Fertigstellung auszuzahlen
$kosten(t,n,x,y) \mapsto n$	$:\Leftrightarrow$	Kosten für Agenten x für den Transport von n Einheiten des Skilltyp t an die Baustelle y.

Im folgenden werden wir die σ_{CSGS}-Struktur $\mathcal{S}$ als ein Setting bezeichnen. Die Modellklasse M_{CSGS} steht für die Gesamtheit an validen Settings die zudem folgende Bedingungen erfüllt:

$$Agent \cap Baustelle = \emptyset \tag{1}$$

Beispiel 1:
Ein Beispiel für eine valides Setting ist die Struktur $\mathcal{S} \in M_{CSGS}$ mit:

$Agent^{\mathcal{S}} := \{a_1, a_2, a_3\}$	$Baustelle^{\mathcal{S}} := \{b_1, b_2\}$
$supply^{\mathcal{S}}(a_1, t_1) \mapsto 2$	$demand^{\mathcal{S}}(b_1, t_1) \mapsto 10$
$supply^{\mathcal{S}}(a_1, t_2) \mapsto 7$	$demand^{\mathcal{S}}(b_1, t_2) \mapsto 5$
$supply^{\mathcal{S}}(a_2, t_1) \mapsto 3$	$demand^{\mathcal{S}}(b_2, t_1) \mapsto 2$
$supply^{\mathcal{S}}(a_2, t_2) \mapsto 5$	$demand^{\mathcal{S}}(b_2, t_2) \mapsto 2$
$supply^{\mathcal{S}}(a_3, t_1) \mapsto 20$	$budget^{\mathcal{S}}(b_1) \mapsto 10$
$supply^{\mathcal{S}}(a_3, t_2) \mapsto 5$	$budget^{\mathcal{S}}(b_2) \mapsto 3$

Definition 3.2 (CSG). Eine Coalitional Skill Game-Signatur (CSG-Signatur) sei definiert als

$$\sigma_{CSG} := \sigma_{CSGS} \cup \{m_{/3}, v_{/2}\}$$

Intuition

$m(x, t, y) \mapsto n$	$:\Leftrightarrow$	Agent x sendet n Einheiten des Skilltyps t an die Baustelle y
$v(x, y) \mapsto n$	$:\Leftrightarrow$	Agent x erhält von Baustelle y die Vergütung n

Wir bezeichnen eine σ_{CSG}-Struktur als ein Game $\mathcal{G}$. Intuitiv ist ein Game ein valides Setting mit einer möglichen validen Lösung: eine mögliche Verteilung der verfügbaren Skills der Baufirmen mit entsprechender Vergütungen der Baustellen.
M_CSG bezeichnen wir als Klasse aller valider Games, die folgende (informelle) Eigenschaften erfüllen:

(1) Ein Agent kann nicht mehr Einheiten eines Skills ausgeben als er besitzt.
(2) Eine Baustelle kann nicht mehr Geld ausgeben, als sie besitzt.
(3) Eine Baustelle gibt genau dann Geld aus, wenn sie vollständig mit den benötigten Skills versorgt wird.

Sei $M_{OCSG} \subset M_{CSG}$ eine Modellklasse die zusätzlich "optimal" ist:

(1) Es exestiert kein Matching, so dass der summierte Gewinn für alle Agenten größer währe.

Sei $M_{FOCSG} \subset M_{OCSG}$ eine Modellklasse, die zusätzlich "fair" ist:

TODO: verweis für Kritärien für "fearheit"

3.2 Terminologie

Definition 3.3 (Koalition). Eine **Koalition** ist ein Zusammenschluss an Agenten die vollständige Informationen über alle Koalitionsteilnehmer besitzen und die Verteilung gemeinsamer Ressourcen auf Bauaufträge beabsichtigen. Ebenfalls haben sie einen gemeinsamen Mechanismus welches den Erlös auf die Koalitionsteilnehmer aufteilt. Formal: $K \subseteq Agenten$ mit folgenden Eigenschaften:

$$\exists x \exists t \exists y. m(x, t, y) > 0 \land K(x) \rightarrow (\forall x' \forall t' m(x', t', y) > 0 \Rightarrow K(x')) \tag{2}$$

$$\exists x \exists y v(x, y) > 0 \land K(x) \rightarrow (\forall x'. v(x', y) > 0 \rightarrow K(x')) \tag{3}$$

Dabei sagt Bedingung 2 aus, dass eine Baustelle nur von einer Koalition gebaut werden kann, sowie Bedingung 3, dass eine Baustelle nur geld an eine Koalition senden kann.

Definition 3.4 (Matching). Ein **Matching** ist eine Konkrete zuordnung von Skillkapazitäten zu Bauaufträgen. Formal das Prädikat $m \in \sigma_{CSG}$

3.3 Mechanismen

TODO: Mechanismuskritätien

TODO: phasen

TODO: überarbeiten

Bei der gegebenen Aufgabenstellung interessieren wir uns für ein Mechanismus, der bei einem gegebenen Setting ein möglichst optimales Game dezentral hervorbringt. Formal gesehen:

$$m : M_{CSGS} \rightarrow M_{CSG} \tag{4}$$

TODO: define mechnismus kritärien

Die Kriterien, nach denen der dezentrale Mechanismus arbeitet und das Ergebnis – bei uns ein Game – bewertet wird, ist nicht hart formuliert und lässt uns Gestaltungspielraum. Die Forderung an das Ergebnis ist, dass die ”soziale Wohlfahrt maximiert wird und die Gewinne möglichst stabil und fair verteilt werde”.

4 ERGEBNISSE

4.1 Theoretische Ergebnisse

4.1.1 Superadditivität.

Lemma 4.1 (Superadditivität). *Das CSG ist superadditiv.*

Bevor wird die Superadditivität des CSG zeigen, füren wir zunächst zwei weitere ”simple” Signaturen mit dazugehörigen Modellklassen ein und zeigen, dass Ergebnisse auf ihnen direkt auf CSG übertragen werden können. Außerdem führen wir die Begriffe *potentielle Matches* einer Koalition, die Menge der *gleichzeitig möglichen Baustellen einer Koalition*, *Outcome-Menge* von zwei Koalitionen und den *Wert* einer Baustelle ein.

(1) **SCSGS** - Simple Coalition Skill Game Setting
(2) **SCSG** - Simple Coalition Skill Game

Wir zeigen später, dass sich jedes Setting bzw. Game zu einem Setting bzw. Game vereinfachen lässt, in dem jeder Agent nur eine Einheit eines Skilltypen besitzt. Diese Vereinfachung erleichtert uns den Beweis der Superadditivität und die Betrachtung möglicher Algorithmen zur Verteilungsberechnung. Hierfür benötigen wir jedoch die neue Relation $AgentOwner_{/2}$, mit der wir uns die Zuteilung einer Skillkapazität zu seinem ursprünglichen Agenten merken:

Definition 4.2 (SCSGS). Eine Simple Coalition Skill Game Setting-Signatur sei definiert als

$$\sigma_{SCSGS} :=$$
$$\{gentOwner_2, Agent_{/1}, Baustelle_{/1}, supply_{/2}, demand_{/2}, budget_{/1}, kosten_{/4}\}$$

Intuition

$AgentOwner(a, x)$	$:\Leftrightarrow$	Agent x gehört zu dem ursprüngl. Agenten a
$Agent(x)$	$:\Leftrightarrow$	x ist ein Agent (Baufirma)
$Baustelle(x)$	$:\Leftrightarrow$	x ist eine Baustelle
$type(x) \mapsto t$	$:\Leftrightarrow$	Agent x ist vom Skilltyp t
$demand(x, t) \mapsto n$	$:\Leftrightarrow$	Baustelle x benötigt n Einheiten vom Skilltyp t
$budget(x) \mapsto n$	$:\Leftrightarrow$	Baustelle x ist bereit einen Gewinn von maximal n bei Fertigstellung auszuzahlen
$kosten(t, n, a, y) \mapsto n$	$:\Leftrightarrow$	Kosten für den zugehörigen ursprüngl. Agenten a für den Transport von n Einheiten des Skilltyp t an die Baustelle y.

Wir bezeichnen eine σ_{SCSGS}-Struktur $\mathcal{SS}$ als *Simple Setting* und die dazugehörige Modellklasse valider Strukturen M_{SCSGS}. Auch hier werden wir die Validitätskriterien nicht weiter erläutern.

Definition 4.3 (SCSG). Eine Simple Coalition Skill Game-Signatur (SCSG) sei definiert als

$$\sigma_{SCSG} := \sigma_{SCSGS} \cup \{M_{/3}, v_{/2}\}$$

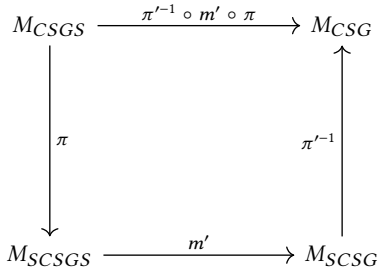
Intuition

$m(x, t, y)$	$:\Leftrightarrow$	Agent x sendet eine Einheiten des Skilltyps t an die Baustelle y
$v(x, y) \mapsto n$	$:\Leftrightarrow$	Agent x erhält von Baustelle y die Vergütung n

Wir bezeichnen eine σ_{SCSG}-Struktur $\mathcal{SG}$ als *Simple Game* mit der dazugehörigen validen Modelklasse M_{SCSG}. Validitätskriterien werden an dieser Stelle nicht weiter betrachtet.

4.2 Beziehungen zwischen den Modellklassen

Die nachfolgenden Betrachtungen wollen wir jedoch auf einer vereinfachten Strukturen anstellen: Bei dieser besitzt jeder Agent nur eine Einheit eines Skilltyps. Um dennoch Aussagen über das CSG machen zu können, zeigen wir, dass sich jedes Setting bzw. Game zu einem Simple Setting bzw. Simple Game überführen lässt. Das erlaubt uns die Ergebnisse von theoretische Betrachtungen von Mechanismen, die bei einem Simple Setting ein Simple Game berechnen, auf das CSG zu beziehen.
Formal:



Dabei müssen folgende Eigenschaften gelten:

$$\pi \quad - \quad \text{total, injektiv} \tag{5}$$
$$\pi^{-1} \quad - \quad \text{surjektiv} \tag{6}$$
$$\pi' \quad - \quad \text{total, injektiv} \tag{7}$$
$$\pi'^{-1} \quad - \quad \text{surjektiv} \tag{8}$$
$$\pi^{-1} \circ \pi \quad = \quad id_{M_{CSGS}} \tag{9}$$
$$\pi'^{-1} \circ \pi' \quad = \quad id_{M_{CSG}} \tag{10}$$

Weiter werden wir nur Mechanismen betrachten die, am Setting keine Änderungen vornehmen, sondern nur Matchings und die Vergütungsverteilung bestimmen. Auf eine ausführliche Definition der gesuchten Funktionen wird hier ebenfalls verzichtet, stattdessen wird eine Intuition gegeben: Ein Setting bzw. Game lässt sich in ein Simple Setting bzw. Simple Game überführen, indem jede Einheit eines Skilltyps eines Agenten als eigenständiger Agent betrachtet wird.
Dabei wird die Zugehörigkeit von Agenten zum Skill in der *AgentOwner* Relation gesichert. Die Kostenfunktion bleibt bestehen und wird lediglich auf den *AgentOwner* übertragen. Bei den Matches und Vergütungen wird ähnlich verfahren. Hierdurch gehen keine Informationen verloren und alle Betrachtungen können auf den vereinfachten Modellen durchgeführt werden.

Definition 4.4 (Potentielle Matches einer Koalition). Sei $K \subseteq Agent$ eine beliebige Koalition. Für K definieren wir die *Menge der potentiellen Matches* $PM(K)$ als

$$PM(K) := \{ \tag{11}$$
$$m : Agent \times Skilltyp \times Baustelle \to \mathbb{N} \mid \tag{12}$$
$$\phi_{Match}(m) \wedge \tag{13}$$
$$\forall x \; \forall t \; \forall y : m(x,t,y) > 0 \to x \in K \qquad \} \tag{14}$$

Dabei bezeichnet $\phi_{Match}(m)$ die Menge der validen Matches:

$$\phi_{Match}(m) := \forall x \; \forall t \; : \left(\sum_{y \in Baustelle} m(x,t,y) \right) \leq \; supply(x,t) \tag{15}$$

Der Ausdruck (14) verdeutlicht, dass nur solche Matches für eine Koalition betrachtet werden, deren liefernder Agent auch Teil der Koalition ist.

Definition 4.5 (Gleichzeitig mögliche Baustellen einer Koalition). Sei $K \subseteq Agent$ eine beliebige Koalition. Die Menge der möglichen Baustellen, die von K gleichzeitig gebaut werden können – $B(K)$ – ist definiert als

$$B(K) := \{ \quad B_k \subseteq Baustelle \mid \tag{16}$$
$$\exists m \in PM(K), \; \forall b \in B_k, \; \forall t : \tag{17}$$
$$\left(\sum_{x \in K} m(x,t,b) \right) \geq demand(b,t) \quad \} \tag{18}$$

Da wir ein Simple Game betrachten, Können wir o.B.d.A. annehmen, dass eine Agent nur einer Koalition zugeordnet ist:

$$K \neq S \Rightarrow K \cap S = \emptyset \tag{19}$$

Hieraus folgt, dass eine Baustelle in einem Spiel nur von einer Koalition gebaut werden kann, da eine Baustelle pro Spiel nur einmal gebaut wird. Um dies formal festzuhalten, definieren wir im Folgenden die Outcome-Relation:

Definition 4.6 (Outcome-Menge zweier Koalitionen). Seien $K, S \subseteq Agent$ beliebige Koalitionen. Eine *Outcome-Menge* weist beiden Koalitionen ein Tupel der Baustellen zu, die sie jeweils zur gleichen Zeit bauen können:

$$Outcome(K,S) := \{ \ (B_K, B_S) \subseteq Baustelle \times Baustelle \mid \tag{20}$$

$$B_K \in B(K) \wedge B_S \in B(S) \wedge B_K \cap B_S = \emptyset \ \} \tag{21}$$

Zur weiteren Vereinfachung definieren wir noch eine Bewertungsfunktion für eine Menge von Baustellen:

Definition 4.7 (Wert einer Baustelle). Sei $B \subseteq Baustelle$ beliebig. Dann ist der *Wert* von B definiert als

$$v(B) := \sum_{b \in B} budget(b) \tag{22}$$

Definition 4.8 (erzielbarer Wert zweier Koalitionen). Seien $K, S \subseteq Agent$ beliebige Koalitionen. Der *erzielbare Wert zweier Koalitionen* K und S ist nun definiert als

$$v(K) + v(S) := \max_{(B_K, B_S) \in Outcome(K,S)} (\ v(B_K) - \varphi_{kosten}(K, B_K) + \tag{23}$$

$$v(B_S) - \varphi_{kosten}(S, B_S) \quad) \tag{24}$$

Die Baustellen, die von einer Koalition gebaut werden, können nicht mehr von einer anderen Koalition gebaut werden. Deshalb müssen wir diese zeitgleich betrachten. $\varphi_{kosten}(B, K)$ ist dabei diejenige Funktion, die basierend auf der vorhandenen Kostenfunktion, der Koalition und den Baustellen die Kosten der Koalition für die Bereitstellung aller notwendigen Skills berechnet, um die Baustellen fertig zu stellen. **Bemerkung**: Nach Definition ist das Ergebnis pareto-effizient.

Existiert eine Koalition isoliert, lässt sich diese auch isoliert betrachten:

Definition 4.9. Sei $K \subseteq Agent$ eine beliebige Koalition.

$$v(K) := \max_{B' \in B(K)} (\ v(B') - \varphi_{kosten}(B', K) \ ) \tag{25}$$

Insbesondere gilt dann auch für zwei allein agierende Koalitionen $K, S \subseteq Agent$:

$$v(K \cup S) = \max_{B' \in B(K \cup S)} (\ v(B') - \varphi_{kosten}(B', K) \ ) \tag{26}$$

Beachte auch das für einen Outcome zweier Koalitionen eine mögliche Baustellenmenge der potentiellen Baustellen der Vereinigung der Koalitionen existiert die alle Baustellen des separaten Outcomes beinhaltet:

$$\forall (B_K, B_S) \in Outcome(K,S), \ \exists B' \in B(K \cup S) : \ B_K \cup B_S \subseteq B' \tag{27}$$

Hieraus können wir die Superadditivität des Spiels schließen:

$$v(K \cup S) \geq v(K) + v(S) \tag{28}$$

Insbesondere ist das Spiel Konvex. Nach einem Theorem (TODO) ist bei einem konvexen Spiel das Shapley Value im Core enthalten, sodass die beste Lösung dieses Spiels die große Koalition ist und eine stabile und faire Lösung immer existiert.

4.2.1 NP-härte des Problems. Im folgenden möchten wir die NP-härte eines gewünschten mechanismusses Zeigen. Hierfür reduzieren wir das Rucksackproblem, ein allgemein bekanntes NP-vollständiges Problem, auf ein CSG. Das Rucksackproblem ist folgendermaßen definiert:

LEMMA 4.10. *Sei $m : CSGS \to CSG$ ein Mechanismus, der ein optimales Matching berechnet, dann ist m NP-hart.*

Definition 4.11 (Rucksackproblem). Sei U eine Menge von Objekten, w eine Gewichtsfunktion und v eine Nutzenfunktion:

$$w : U \to \mathbb{R} \tag{29}$$

$$v : U \to \mathbb{R} \tag{30}$$

Sei weiter $B \in \mathbb{R}$ eine vorgegebene Gewichtsschranke. Gesucht ist eine Teilmenge $K \subseteq U$, die folgende Bedingungen erfüllt:

$$\sum_{w \in K} w(u) \leq B \tag{31}$$

$$max(\sum_{w_i nK} v(u)) \tag{32}$$

Sei $K = \{U, w, v\} \in KNAPSACK$ ein beliebiges Rucksackproblem. Wir geben eine übersetzung in eine CSGS-Signatur an:

$$Agent = \{a\} \tag{33}$$

$$supply(a, t) \mapsto B \tag{34}$$

$$Baustelle = U \tag{35}$$

$$demand(u, t) \mapsto w(u) \tag{36}$$

$$kosten(t, n, x, y) \mapsto 0 \tag{37}$$

TODO: In worten beschreiben was hier passiert.

Intuitiv sorgt die Bedingung 31 dafür, dass der Agent seine resourcen nicht überschreitet, sowie die Bedingung 32, dass der Agent sein profiet maximiert. Beides sind ebenfalls Anforderungen, die von einem Mechanismus erfüllt werden müssen der ein optimales CSG berechnet.

TODO: referenz zum abschnit über den mechanismus

Angenommen es gäbe ein $m \in P$ der aus einem CSGS ein CSG mit einem optimalem matching berechnet, dann währe auch $KNAPSACK \in P$. Dieses ist jedoch ein wiederspruch zur annahme $KNAPSACK \in NP$.

4.3 Praktische Ergebnisse

5 ZUSAMMENFASSUNG

TODO: Zusammenfassung