

A Polynomial Kernel-Oriented Coalition Algorithm for Rational Information Agents*

Matthias Klusch and Onn Shehory
Institut für Informatik,
Christian-Albrechts-University Kiel,
24118 Kiel, Germany

e-mail: *mkl@informatik.uni-kiel.d400.de*

Department of Mathematics & Computer Science,
Bar Ilan University,
Ramat Gan, 52900 Israel

e-mail: *shechory@bimacs.cs.biu.ac.il*

Abstract

Information agents behave like active intelligent front-ends of stand-alone information systems. The main purpose of such an agent is to gather intensionally relevant information in non-local domains. They may either work as individuals or efficiently cooperate in order to satisfy their own set of given information search tasks. However, in particular the need to respect the database autonomy requirements and to cope with semantic heterogeneity hinders such a cooperation. In this paper we present a solution for handling the autonomy during decentralized information-gathering by rational cooperation. For this purpose, methods for terminological knowledge representation and inference, as well as for utilitarian coalition formation among the information agents, are used. The decentralized agent-utility calculation bases on the agent's productions, resulting from executing tasks of finding dependencies between local terminological information models. There is no prior need nor a possibility to browse through foreign database schemas in order to find some possibly relevant data. The coalition algorithm proposed in this paper enables efficient cooperation via the formation of Kernel-oriented stable coalitions among rationally cooperating information agents in polynomial time.

Topic Areas:

Game-Theoretic Approaches, Cooperative Information Systems, Information Gathering

*The work reported here is supported in part by the NSF, grant No. IRI-9423967 and the Israeli Ministry of Science, grant No. 6288. We thank Prof. Dieter Klusch, Prof. Peter Kandzia and Prof. Sarit Kraus for giving support by many helpfully hints and advices.

1 Introduction

The increase in the amount of information in the last decades has caused the formation of multiple heterogeneous, autonomous databases in a decentralized environment as well as their global interconnections. As a result, the search for semantically relevant, non-local data has become an exhaustive procedure, either for a human user or for a computerized data-search server. Several solutions to this information-gathering problem by different types of agent-based systems are provided e.g., in [1], in the InfoSleuth project [8] or in [5]. In addition, there is also a recent trend to create mobile software agents capable for an user-dependent information search in the WWW, e.g. [9].

These approaches are essentially different from that of rationally cooperating information agents in a decentralized environment introduced in [12]. The latter solves the problem of recognizing intensional data dependencies between stand-alone databases while respecting their autonomy requirements by the use of terminological knowledge representation and inference, as well as methods for utilitarian coalition formation among autonomous agents. The coalition formation model proposed in [16] leads to individually rational solutions but does not guarantee stability in a coalition-theoretical sense (cf.Sect.3,[10]). Therefore, in this paper we present an algorithm which enables utilitarian coalition formation for rationally interacting information agents in general types of environments with a guaranteed stability, reachable in polynomial time. We base our solution on the Kernel stability concept from game theory[10] as done in [22]. The coalition model presented in the latter is inappropriate for the specific case of information agents which shall solve the problem of rational information-gathering among several decentralized and autonomous databases. Hence, we provide such a solution by a new Kernel-oriented coalition algorithm in terms of specific functions and procedures to be performed by each information agent locally, state its complexity and discuss its essential properties.

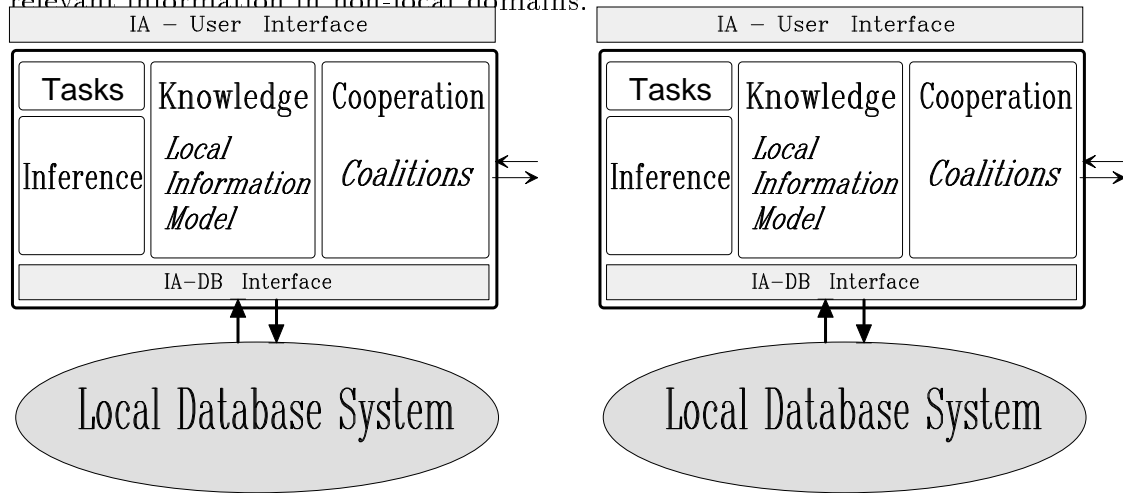
2 Information Agents

In the literature there can be found many different views on what an agent is; the answer mostly depends on who poses the question. This is also the case for the term information agent. In [27] it is defined as an agent which has access to at least one, and potentially many information sources, and is able to collate and manipulate information obtained from these sources to answer queries posed by users and other information agents. Others restrict the term to a more linguistic-based negotiation between knowledge-based entities which relies on a presumed common ontology, e.g. [8, 25].

In our context an information agent is an active, intelligent database front-end trying to satisfy its own application-specific tasks, alone or in cooperation with others, for information gathering. The federative agent system FCSI ¹ is constituted by a set of cooperative

¹FCSI is a shortcut for '*F*ederative *C*ell System for discovery of *I*nterdatabase dependencies'.

information agents (cf. Fig. 1) which try to rationally coalesce to discover intensionally relevant information in non-local domains.



For this purpose each FCSI agent builds its own local terminological information model LIM. Such an information model contains a more linguistic-based description of semantic aspects of selectively exportable schema data². Each agent is able to detect several kinds of interrelation knowledge concerning these schema data locally³ ([12, 15]).

During the coalition formation process each agent rationally decides to coalesce with other information agents by calculating its own utility on the set of discovered interdatabase dependencies which e.g. satisfy own and/or received search tasks. These utilities and respective coalition values can be seen as a measure of how close the information agents' domains are related (cf. Sect. 5). Based on such an interrelation knowledge, each agent can pose directed, intensional queries on non-local data only to committed members of the same coalition. We will focus on the coalition formation among the FCSI agents.

3 Game-Theoretic Definitions and Concepts

Below we provide the definitions of game-theoretic concepts which are necessary to understand our coalition formation process.

Definition 3.1: *Cooperative Game* (\mathcal{A}, v)

Let \mathcal{A} be a set of agents $a_i, i \in \{1, \dots, n\}$ and $C \subseteq \mathcal{A}$ be a coalition.

1. *Characteristic function* (or coalition value function) of the game (\mathcal{A}, v) assigns to each coalition C a value which measures its utility achievable as a whole by cooperation among its members: $v : \mathcal{P}(\mathcal{A}) \mapsto \mathcal{R}^+, v(\emptyset) := 0$.

²For details on the FCSI information agents see e.g. [15]. Some related works are [2, 8] and the LICS/LIKE project (see e.g. [3]).

³This includes the identification of so-called interdatabase dependencies which are comparable e.g. to the boolean-valued data dependency predicate in [23].

2. $v(\{a_i\})$ denotes the *self value* of a single-agent coalition.
3. Each partition of \mathcal{A} (a set of non-empty, mutually disjunctive subsets $C \subseteq \mathcal{A}$) is called a *coalition structure* \mathcal{C} , $|\mathcal{C}| = m \leq n$.
4. A *payment configuration* $PC = (u, \mathcal{C})$ for the coalition structure \mathcal{C} denotes a *payoff distribution* $u : P(\mathcal{A}) \mapsto R$ of coalition values in (\mathcal{A}, v) to each agent: $(u_1, \dots, u_n; C_1, \dots, C_m)$, $u_i \equiv u(\{a_i\})$, $u(R) := \sum_{a_i \in R} u_i$ with $u(C_k) = \sum_{a_i \in C_k} v(C_k), \forall C_k \in \mathcal{C}$ (Efficiency).

Main assumptions for a game (\mathcal{A}, v) :

- each $C \subseteq \mathcal{A}$ is given a value $v(C)$.
- a payment configuration represents one solution of the game.
- utility units are comparable and transferable among the agents. \square

Definition 3.2: *Super-additive vs. Non-super-additive, cooperative games*

Let \mathcal{A} be a set of agents $a_i, i \in \{1, \dots, n\}$, (\mathcal{A}, v) a cooperative game.

(\mathcal{A}, v) is *super-additive* $:\Leftrightarrow \forall C_k, C_l \subseteq \mathcal{A} : v(C_k \cup C_l) \geq v(C_k) + v(C_l)$

(\mathcal{A}, v) is *coalitional rational* $:\Leftrightarrow \forall C \subseteq \mathcal{A}, a \in \mathcal{A} : a \in C \Leftrightarrow v(C \cup \{a\}) \geq v(C)$

(\mathcal{A}, v) is *non-super-additive* $:\Leftrightarrow \exists C_k, C_l \subseteq \mathcal{A} : v(C_k \cup C_l) < v(C_k) + v(C_l)$

(\mathcal{A}, v) symmetric game $:\Leftrightarrow \forall a_i, a_j \in \mathcal{A} (i \neq j)$ symmetric

a_i, a_j symmetric agents $:\Leftrightarrow \forall C_k \subset \mathcal{A}, a_i, a_j \notin C_k : v(C_k \cup \{a_i\}) = v(C_k \cup \{a_j\})$. \square

Definition 3.3: *Properties of a Payment Configuration PC*

Let $PC = (u, \mathcal{C})$ be a payment configuration for a coalition structure \mathcal{C} .

1. Rationality of the PC :

Individually rational $\quad \forall a_i \in \mathcal{A} : u_i \geq v(\{a_i\})$

Group rational $\quad u(\mathcal{A}) = v(\mathcal{A})$

Coalition rational $\quad \forall T \subseteq \mathcal{A} : u(T) \geq v(T)$

Locally coalition rational $\quad \forall R \subseteq C_k \in \mathcal{C} : u(R) \geq v(R)$

2. Pareto-Optimality of the PC :

pareto-optimal: $\neg \exists u' : \forall i \in \{1, \dots, n\} u'_i \geq u_i$

locally pareto-optimal in $L = \{u^{(1)}, \dots, u^{(k)}\}$: $\neg \exists u' \in L : \forall i \in \{1, \dots, n\} u'_i \geq u_i$. \square

A stable solution of a cooperative game (\mathcal{A}, v) is denoted by a set of payment configurations PC which are atleast individually rational. The meaning of the notion of stability depends on the particular coalition theory it is associated with. Well-known notions of utility division stability are e.g. those of the Core **C**, Stable Set **S**, Bargaining Set **M** and the Kernel **K**. A calculated solution of a given game is called CT -stable ($CT \in \{\mathbf{C}, \mathbf{S}, \mathbf{M}, \mathbf{K}\}$), if all its PC s are stable in a sense given by the actually considered coalition theory CT . Thereby, each coalition theory defines the possible solution space for a cooperative game. For an introductory survey of them refer to e.g. [10, 22].

Since the present work provides a Kernel-oriented coalition algorithm we give the definition of the Kernel \mathbf{K} as well as the respective basic calculation scheme, the modified transfer scheme for a \mathbf{K} -stable PC, and omit the other coalition theories.

The Kernel [4] is a set of payment configurations PC in which the coalition structure \mathcal{C} is stable in the sense that there is an equilibrium between all pairs of individual agents which are in the same coalition, i.e. they do not dominate each other in such a PC . It leads symmetric agents to receive equal payoffs.

Definition 3.4: *The Kernel \mathbf{K}*

The set of payment configurations of a cooperative game, where each pair of agent is in an equilibrium is called *Kernel*

$\mathbf{K} := \{(u, \mathcal{C}) \mid \forall a_k, a_l \in C \in \mathcal{C} : (a_k, a_l) \text{ in equilibrium}\}$, with

1. *Excess of a coalition* $C \notin \mathcal{C}$ with respect to an utility distribution u in (u, \mathcal{C}) :
 $e(C, u) := v(C) - u(C)$.
2. *Surplus*, or strength, s_{kl} of agent a_k over agent a_l in the same coalition $(a_k, a_l \in C \in \mathcal{C})$ with respect to a payment configuration (u, \mathcal{C}) : $s_{kl} := \max_{R \notin \mathcal{C}, a_k \in R, a_l \notin R} e(R, u)$.
3. Agent a_k *dominates* agent a_l in the same coalition $(a_k, a_l \in C \in \mathcal{C})$ with respect to a payment configuration (u, \mathcal{C}) iff $s_{kl} > s_{lk}$ and $u_l > v(\{a_l\})$.
4. A pair of agents $a_i, a_k \in C \in \mathcal{C}$ is in *equilibrium* iff
 $(s_{kl} = s_{lk}) \vee (s_{kl} > s_{lk} \wedge u_l = v(\{a_l\})) \vee (s_{kl} < s_{lk} \wedge u_k = v(\{a_k\}))$. \square

Definition 3.5: *Modified Transfer Scheme for \mathbf{K} -stable PC*

1. Each sequence $(u^{(1)}, \mathcal{C}), \dots, (u^{(i)}, \mathcal{C}), \dots$ of payment configurations transferring in each step i positive sidepayments α among the agents a_k, a_l by the following assignments

$$(u^{(i+1)}, \mathcal{C}) : \begin{cases} u_k^{(i+1)} & := u_k^{(i)} + \alpha \\ u_l^{(i+1)} & := u_l^{(i)} - \alpha \\ u_j^{(i+1)}, (j \neq k, l) & := u_j^{(i)} \end{cases}$$

is called a *Transfer Scheme*. The calculation of α is determined by the considered coalition theory, means the notion of stability. A *convergent transfer scheme for \mathbf{K} -stable PCs* iteratively computes for a given coalition structure \mathcal{C} and an initial payment configuration $PC^0 = (u, \mathcal{C})$ another payment configuration (u', \mathcal{C}) which is in the Kernel \mathbf{K} .

2. The convergent Transfer Scheme for \mathbf{K} -stable PC [24] is determined by the mutual *demand* d_{kl} for each pair of agents a_k, a_l as an upper bound for sidepayments α .

Let $PC = (u, \mathcal{C})$, s_{kl} surplus of agent a_k over agent a_l in PC , $a_k, a_l \in C \in \mathcal{C}$, $\alpha \leq d_{kl}$ with

$$d_{kl} := \begin{cases} \min\{(s_{kl} - s_{lk})/2, u_l\} & \text{if } s_{kl} > s_{lk} \\ 0 & \text{else} \end{cases}$$

3. *Modified Transfer Scheme* for the Kernel \mathbf{K} :

Calculation of an ε -approximately \mathbf{K} -stable payment configuration $PC = (u, \mathcal{C})$ by terminating Stearns transfer scheme⁴ for \mathbf{K} -stable PC iff the *relative PC-kernel error* $re(u)$ is sufficiently small $re(u) := \frac{\max\{s_{ij} - s_{ji} \mid a_i, a_j \in C \in \mathcal{C}\}}{u(\mathcal{A})} \leq \varepsilon$. \square

There exist also well-known transfer schemes for the Core and the Bargaining Set in game-theory literature, but no algorithm for calculating a stable payment configuration *and* concurrently forming the respective coalition structures⁵. Thus, most of the game theory work only predicts, given a coalition configuration, how the players (or agents) will distribute the benefits. They do not provide them with an algorithm for the formation of coalitions.

Development of regulations and strategies in an abstract manner for coalition formation in various environments appear in the area of Distributed Artificial Intelligence (DAI), e.g., [28, 20, 22, 21, 11]. Recently, interdisciplinary research started to investigate how to adapt such proposals, if possible, to ones which are appropriate also for application-specific environments, like in [6, 12, 15, 16].

4 Coalition Formation in DAI

Distributed artificial intelligence is concerned with cases in which multiple agents interact in order to achieve goals. Several solutions to the coalition formation problem have been suggested by researchers in the field of DAI. Some solutions concentrate on the special case of autonomous agents in a super-additive environment [20, 11, 28], provided for coalition formation in Multi-Agent Systems (MAS), where each agent tries to increase its own personal utility via cooperation. Other MAS solutions for general environments are e.g. given in [22] and in [19]. The latter presents a coalition formation model for bounded-rational agents, a general classification of coalition games and allows for varying coalitional values where the value of a coalition depends on the computation time. These solutions strongly base on the game theory concepts. However, some coalition formation algorithms in DAI are based on operations research, combinatorial algorithms and graph theory [21]. These are most appropriate for coalition formation in Distributed Problem Solving (DPS) cases, where the agents cooperate in order to increase the outcome of the whole system.

In the game theoretic approach (which we adopt), given a payment configuration PC one is interested in building various coalition structures that have an utility distribution which is stable with respect to the considered coalition theory. Thus, game-theory prone coalition formation algorithms for multiple agents may possess of the following properties

- be computation- or negotiation-oriented.
- be centralized or decentralized.

⁴The truncated transfer scheme still guarantees convergence since the α -reduction process which leads to convergence is not modified.

⁵Note that each transfer scheme in game-theory computes a payoff vector for a given, *fixed* coalition structure \mathcal{C} .

- terminate with a polynomial computational complexity.
- provide a stable payment configuration, optionally pareto-optimal.
- determine behaviour and decision-making process of each coalition unit.

Our algorithm is negotiation-oriented, decentralized, terminates within a polynomial time, results in a stable payment configuration, and provides the single, rational information agent with methods for behavior and decision making.

5 Coalitions of Information Agents

For coalition formation among the rational information agents we adapted the production-oriented view of [20] for utilitarian coalitions to the FCSI agents' domain. The agents' utility functions are calculated with respect to the utility from their own knowledge-based productions, obtained by execution of received or own information search tasks. The utility an agent obtains when it executes its own search tasks exclusively by itself is denoted as the agent's self-value. The value of a coalition is the sum of utilities all its members calculate for their productions. We assume that the agents know of the utility functions, agree on the utility division method and that side-payments are transferable among the agents. The used coalition algorithm (cf. Sect. 5.1) converges to individually rational and Kernel-oriented stable coalition structures (cf. Def.3.1)⁶.

There is one important negotiation constraint to be satisfied by all information agents, the so-called information availability convention for coalitions (IAC). The IAC requires that all members of a fixed coalition are mutually committed to provide the availability of their local schema information they used for building the value of their coalition. Consequently, each search task sent by one information agent to another implies an access-right for the latter to the respective local schema data of the first, if they will be in the same fixed coalition⁷. Thus, during negotiation the productions and value of each potential coalition must be projected to the amount which is contributable to other coalition entities.

Definition 5.1: *Productions, agent utility, coalition value, contributable amount*

Let \mathcal{A}_{FCSI} be a set of n FCSI information agents, $a_i, a_k, a_j \in \mathcal{A}_{FCSI}$, $C \subseteq \mathcal{A}_{FCSI}$. Let further be

1. $p(t_{tid_x}^{a_k, a_j})$ the knowledge-based production of agent a_k for own or received search task t_{tid_x} from agent a_j , i.e. all discovered interdatabase dependencies between a_k and a_j with respect to the search term of task t_{tid_x} using all available informations,
 $Prod_{a_k}^s$ self-productions for own search tasks restricted on local information model,
 $Prod_{a_k, C}$ set of a_k 's productions for search tasks received from agents of coalition C ,

⁶Some coalition algorithms which use the so-called bilateral Shapley-Value[11] as a fair and individually rational division method were provided in earlier work ([12, 13, 16]).

⁷Note that an exportable local schema data is not necessarily available for all other agents and there may exist terminological descriptions of local data which are not considered as search task terms. In addition, every information-search task has to be selectively broadcasted in the FCSI.

$Prod_{a_k, C}[S]$ set of parts of productions $p \in Prod_{a_k, C}$ which are available for all members of coalition S ,

$Prod_{a_k, C}^{(C)}[S]$ set of parts of productions $p \in Prod_{a_k, C}[S]$ whose elements relate data exclusively within the local domains of C^8 , $Prod_{a_k, C}^{ca} := Prod_{a_k}^s[C] \cup Prod_{a_k, C \setminus \{a_k\}}^{(C)}[C]$.

2. *Agent Utility Function*: $U_{agent, tid}^{type}(p(t_{tid, x, a_j}^{a_k}))$, with production $p(t_{tid, x, a_j}^{a_k})$
3. *Coalition Value* $v(C)$: $\mathcal{P}(\mathcal{A}) \mapsto \mathcal{R}^+$, $v(C) := \sum_{a_k \in C, p \in Prod_{a_k, C}^{ca}} U_k(p)$
4. The *contributable amount of the (self-) coalition value* for (1- or) n-agent coalition entity C to another coalition entity S is defined as $v(C)[S] := \sum_{a_k \in C, p \in Prod_{a_k, C}^{ca}} U_k(p)$. \square

Every calculated polynomial Kernel-stable solution is defined to be *pKS-stable*. Different types of agent utility functions lead to respectively different coalition types. In particular, we consider formation of coalitions relying on quantitative coalition types. There, each agent's utility bases e.g. on the amount of satisfaction of submitted own and received search-tasks (C_{ti}), or only on the amount of satisfaction of all own tasks (C_{otsat}), or beneficially on the amount of satisfaction of received search tasks (C_{tser}). Currently we investigate also some qualitative coalition types concerning a particular degree of task satisfaction.

In general, non-super-additive environments are expected in cases where the increase in the size of the coalition is costly. There, it may be less beneficial to form a joint coalition than to let coalitions remain disjoint. Such costs may arise from internal communication and coordination requirements. This strongly holds for the case of information agents, because any connection to another agent is costly both due to communication costs and due to remote data retrieval costs. Therefore, large coalitions are costly, and the expected utility from satisfying information search tasks may be less than the overhead of internal coalition costs. For the considered coalition value function v (Def. 5.1) it can be proven that there may exist cooperative games (\mathcal{A}_{FCSI}, v) which are non-super-additive⁹.

5.1 A Kernel-Oriented Coalition Algorithm (KCA)

The following coalition algorithm KCA advances the information agents from one coalition structure to another in order to increase the agents' payoff, thus increasing rationally motivated cooperation. In each step, at least one coalition will make an attempt to improve the payoffs of its members by calculating a respective payment configuration as a proposal to another, actually most promising coalition. If both agree on such a proposal it will be broadcasted to the other coalitions. Thereby, these are temporarily enforced to accept the corresponding payoff distribution for all agents. Since this affects in particular agents which were not involved in the bilateral agreement, they may be dissatisfied with their new payoff

⁸This restriction is immediately implied by the IAC convention.

⁹This is mainly caused by the restriction on the contributable amounts of productions for coalition value calculations. Therefore, the coalition environment including Def.5.1 and the IAC is of general type.

and will react with its proposal to the newly formed agent community in the next round. Negotiation continues until all proposals of all coalition entities of the currently valid coalition structure are rejected or a pKS-stable payoff configuration has been reached or a predefined time-period ends.

Algorithm 5.1: KCA

Let \mathcal{A}_{FCSI} set of FCSI information agents, and further be

- $csize_{min}$ and $csize_{max}$ given lower and upper bound of coalition sizes for calculating payment configurations,
- $rec - proposals_{C_i}$ the first-come-first-served (fifo) queue of proposals (C_t, C_i, PC_{ti}) coalition C_i has actually received from all other coalitions C_t ,
- PC^{rnum} confirmed payment configuration of the coalition formation round $rnum$,
- $rtime$ given time-period unit for receiving messages,
- $rec - messages$ the (fifo-)queue of local agent's received messages,
- $preferenceList_i$ list of preferred (1- or m-agent) coalition entities in descendent order for (1- or m-agent) coalition entity C_i ,
- $paccept, rptimeout$ and $ctimeout$ boolean flags which indicate if a payment configuration proposal is accepted by some representative for all agents in its coalition, and if a given time-period for receiving proposals from the agent community in each round and coalition negotiations at all has been exceeded, respectively.
- $GetBroadcastedPC$ boolean flag indicates if a payment configuration broadcasted by some coalition as PC^{rnum} has been received.

$rnum:=0$;

for each agent a_m do in each negotiation round:

▷ $rnum:=rnum + 1$;

 if $rnum > 1$ then $Vote(C_i)$;

▷ if agent a_m is representative of Coalition C_i then perform:

$preferenceList_i := Rank(C_i, \mathcal{P}(\mathcal{A}_{FCSI}) \setminus C_i)$;

$paccept:=FALSE$;

 for each $C_j \in preferenceList_i$ do

 if $csize_{min} \leq |C_j| \leq csize_{max}$ then begin

$GetKVal(C_i, C_j); PC_{ij}^{rnum} := CalculatePC(C_j)$;

 if $EvalPC(C_i, C_j, PC_{ij}^{rnum})$ then $Send(PC_{ij}^{rnum}, C_j)$; end;

 repeat $Wait(rtime)$; $Qmanage(rec - messages, rec - proposals_{C_i})$; until $rptimeout$;

 for each $PC_{ki}^{rnum} \in rec - proposals_{C_i}$ do

 if not $paccept$ then begin

 if $Compare(PC_{ki}^{rnum}, PC_{ik}^{rnum})$ and $Decide(PC_{ki}^{rnum})$ then begin

$accept_{ik}:=TRUE; paccept:=TRUE; Broadcast(paccept, C_k)$; end; end;

```

for each  $C_k \neq C_i$  do
  if GetAccept( $C_k, rec - messages$ ) then begin
     $C_{ik} := \text{FormCoalition}(C_i, C_k, PC_{ki}^{rnum})$ ; SendConfirm( $C_k$ );
    Broadcast( $PC_{ki}^{rnum}$ );  $PC^{rnum} := PC_{ki}^{rnum}$ ; break; end
  else
    if  $accept_{ik} = \text{TRUE}$  then begin
      SendAccept( $C_k$ );
      if GetConfirm( $C_k, rec - messages$ ) then begin
         $C_{ik} := \text{FormCoalition}(C_i, C_k, PC_{ik}^{rnum})$ ;
        Broadcast( $PC_{ik}^{rnum}$ );  $PC^{rnum} := PC_{ik}^{rnum}$ ; break; end; end
      else repeat Wait( $rtime$ ); Qmanage( $rec - messages, rec - proposals_{C_i}$ );
        until GetBroadcastedPC;
    if  $cptimeout$  then CancelNegotiation;

```

Due to space limitations we give a brief, informal description of the used functions.

- **function** Vote(C : Coalition): performs a majority vote as in e.g. [17]; focus future communication with C on its voted representative.
- **function** Rank(C : Coalition; P : Coalition-set) ranks the other coalitions $C_k \in P$ wrt. their contributable amounts $v(C_k \cup C)[C]$ ¹⁰ and different coalition types; local calculation of these values bases on productions for search tasks that were previously received from the C_k agents.
- **function** GetKVal(C_i, C_j : Coalition): get non-local surplus values for coalition C_i from representative of C_j , i.e. $\forall a_k \in C_i, a_l \in C_j$: get s_{lk} (Def.3.1).
- **function** CalculatePC(C : Coalition): calculates a payment configuration proposal using the modified transfer scheme (Def. 3.5).
- **function** EvalPC(C_i, C_j : Coalition; PC_{ij} : Payment configuration): check if the bilateral proposal is better for both than the currently valid payment configuration, i.e. if $\sum_{a_x \in C_i \cup C_j} u_x^{(PC_{ij})} \geq \sum_{a_x \in C_i \cup C_j} u_x^{(PC^{rnum-1})}$ then EvalPC:=TRUE.
- **function** Compare($PC_{ki}^{rnum}, PC_{ik}^{rnum}$: Payment configuration): check if the received proposal PC_{ki}^{rnum} allows every agent $a_l \in \mathcal{A}_{FC SI}$ to gain more benefit u_l than the own one PC_{ik}^{rnum} calculated for C_k (then Compare:=TRUE).
- **function** Decide(PC_{ki}^{rnum} : Payment configuration): check if the received proposal PC_{ki}^{rnum} is better than all other received ones $PC_{si}^{rnum} \in rec - proposals_{C_i}$ by Decide:=Compare($PC_{ki}^{rnum}, PC_{si}^{rnum}$).
- **function** FormCoalition(C_i, C_j : Coalition; PC_{ji} : Payment configuration): forms a new coalition $C = C_i \cup C_j$ wrt the bilateral agreement on the payment configuration PC_{ji} ; this atmost extends coalitions which are already formed in the previous formation round.

¹⁰Note that $v(C \cup S)[S] = v(C \cup S)$ holds, but for the so-called *pure coalition value* $v^*(C) := \sum_{a_k \in C, p \in Prod_{a_k}^S \cup Prod_{a_k, C \setminus \{a_k\}}^{(C)}} U_k(p)$ this is not true (i.p. $v^*(C)[C] = v(C)$).

- `Send`, `GetAccept`, `SendAccept`, `SendConfirm`, `GetConfirm`, `Broadcast` are communication functions: message sending and acceptance. The details are omitted due to simplicity of the functions and space limitations.

5.2 Complexity of the KCA

The number of coalitions to be considered during the KCA is

$$n_{coalitions} = \sum_{i=csize_{min}}^{csize_{max}} \binom{n}{i} = \sum_{i=csize_{min}}^{csize_{max}} \frac{n!}{i!(n-i)!}$$

which is, in the worst case, of order $O(n^{csize_{max}})$. This may be multiplied by the number of coalition types, which is a constant and therefore does not affect the order of the complexity. When the agents compute the set of coalitions, the coalitional values and the coalition structures, the order of complexity is equal to the order of the number of the coalitions.

During the KCA process, an agent is required to compute $O(n_{coalitions})$ coalition values and design $O(n)$ coalition structures. In each iteration round, whenever a coalition structure is considered, a pKS -stable PC shall be calculated. This is done by `CalculatePC`, where the KCA employs a truncated transfer scheme for calculating such PCs. Each transfer scheme iteration step consists of the following parts:

- $n_{coalitions}$ excesses are calculated, each requires $O(n)$ calculations. Among the excesses, the surpluses are searched for. The composite complexity of the excess and surplus calculations is $O(n \times n_{coalitions})$.
- The maximum surplus is located by $O(n_{coalitions})$ operations. Other calculations are of a lower complexity.

Summing up, the complexity of one transfer scheme iteration step is $O(n \times n_{coalitions})$. The resulting payoff vector of the transfer scheme will converge to an element of the polynomial-Kernel (with a relative error not greater than ε) within $n \log_2(e_0/\varepsilon)$ iteration steps [24], where e_0 is the initial PC error and ε is the predefined allowed error¹¹.

An agent will perform the transfer scheme $O(n)$ times per KCA iteration round from which there are in turn $O(n)$. Other procedures performed during the KCA are of lower complexity concerning the functions studied above. Therefore, the complexity of the KCA per agent is $O(n^3 n_{coalitions})$ computations. In addition to the computational operations, some communication operations are required. In general, each agent is required to perform $O(n)$ communication operation per KCA iteration round, and this is done $O(n)$ times. Therefore, the communication complexity per agent is $O(n^2)$.

The most calculation-consuming step concerns the knowledge-based recognition of interrelations. This shall be performed prior to starting the KCA. There, each agent a_i performs $o(n^2)$ computational operations [16].

¹¹Note that the logarithm does not depend on n , and therefore does not affect the order of complexity, although it may contribute a large factor.

5.3 Properties of the KCA

The KCA strongly relies on local computations and respects the IAC requirement to be satisfied by the information agents. In particular, each agent

- receives for local proposal calculation from another possible coalition that excludes it only the surplus values which are not locally computable, or
- locally calculates the amount of coalition values contributable to other coalition entities by itself. Such calculations are restricted to coalitions in which it is a member.

Since, during the formation process, coalitions can vary, following the IAC access to remote schema data is possible only within a fixed coalition at the end of the process. Other coalition algorithms for rational information agents which can also be used for super-additive and not-super-additive environments are in [12, 16]. However, there exist algorithms which could may be adapted to this environment as well. But most of them assume (explicitly or implicitly) complete information for each agent. This is in contradiction with the IAC requirement for information agents. In addition, some stable algorithms seem to be better than e.g. the one presented in [16]. They provide perfect stability wrt the considered coalition theory, but with exponential computation efforts (e.g. the Bargaining Set). In contrast, in the information agents case we need a reduction to a polynomial complexity. This is satisfied with the KCA as proved in the previous section. The KCA is an anytime algorithm, i.e. it will provide the agents with an individually rational, pKS-stable payment configuration and associated coalition structure at any time it terminates, but it does not necessarily lead to a Pareto-optimal payment configuration¹².

In each negotiation round a pKS-stable configuration will be accepted by the agent community. Moreover, the algorithm provides each agent with a regulation method to choose among proposed configurations due to its personal rationality.

Since each agent could modify or delete some of its own search-tasks or create new ones, agents must be able to react on such a changing environment. Varying the known set of information-search tasks in the FCSI causes several reactions by the agents. It will in particular modify the set of discovered interdatabase dependencies, thus may change the agents' utilities. This in turn affects the coalition formation process. There will either be a new attempt to converge to a pKS-stable coalition structure, using the last calculated payment configuration as an initial one for continuing the formation process, or, if such changes in the task set are continuous, there will be infinitely many KCA iterations.

6 Conclusion

We introduced a coalition algorithm which enables utilitarian coalition formation for rationally interacting information agents in general environments with a guaranteed Kernel-oriented stability which is reachable in polynomial time. Such information agents solve the

¹²However, to guarantee this property all possible coalition structures must be considered, i.e. in exponential time. Moreover, for each of them there may exist an infinite number of real-valued payoff distributions. In addition, Pareto-optimal PCs are not necessarily individually rational for a specific agent.

problem of information-gathering among several decentralized and autonomous databases by rational cooperation. The solution is provided by a new coalition algorithm in terms of specific functions and procedures to be performed by each information agent locally. It is a polynomial anytime algorithm and respects the specific requirements the FCSI agents have to deal with. In addition, it can be adapted for an utilitarian, decentralized information-search in the Internet.

Toward an implementation of FCSI information agents an *Interactive Development Environment for the specification and simulation of Agent Systems* IDEAS [15] as well as the knowledge-based component [7] has been implemented on HP-UX and SUN-Solaris workstations. For first simulation results of the basic negotiation schema of the KCA we refer to [22]. Ongoing research mainly focus on further investigations of several coalition types and the integration of uncertainty.

References

- [1] Barbuceanu, M., Fox, M.S., 1994, "The information agent: an infrastructure for collaboration in the integrated enterprise", Proc. SIG Meeting on Cooperating Knowledge-Based Systems CKBS-94, Keele(UK).
- [2] Blanco, J.M. et al., 1994, "Building a federated relational database systems: an approach using a knowledge-based system", Intern. Journal on Intelligent Coop. Inform. Syst. 3(4).
- [3] Burg, J.F.M. et al., 1993, "A data-dictionary as a lexicon: An application of linguistics in information systems", Proceedings of the 2nd International Conference on Information and Knowledge Management.
- [4] Davis, M., and Maschler, M., 1965, "The kernel of a cooperative game", Naval research Logistics Quarterly, 12:223-259.
- [5] Decker, K., Lesser, V. et al., 1995, "MACRON: an architecture for multi-agent cooperative information gathering", Proc. CIKM-95 IIA Workshop Intelligent Information Agents.
- [6] Fischer, K., Müller, J.P., 1996, "A decision-theoretic model for cooperative transportation scheduling", Proc. MAAMAW-96, Eindhoven, LNAI Series Vol. 1038, Springer Verlag.
- [7] Gao, H., 1996, "Implementierung eines terminologischen Wissensrepräsentations- und Inferenzsystems TEWIS", MSc. Thesis, University of Kiel, Germany
- [8] Jacobs, N., Shea, R., 1995, "Carnot and InfoSleuth - Database Technology and the WWW", ACM SIGMOD Intern. Conf. on Management of Data, May 1995.
- [9] Jacobs, N., Shea, R., 1996, "The role of Java in InfoSleuth: Agent-based exploitation of heterogeneous information resources", Proc. of Intranet-96 Java Developers Conference.
- [10] Kahan and Rapoport, 1984, *Theories of coalition formation*, Lawrence Erlbaum, London.

- [11] Ketchpel, S., 1993, "Coalition formation among autonomous agents", Proc. of MAAMAW-93.
- [12] Klusch, M., 1994, "Using a cooperative agent system for a context-based recognition of interdatabase dependencies", Proc. CIKM-94 Workshop on 'Intelligent Information Agents', Gaithersburg (USA).
- [13] Klusch, M., 1995, "Cooperative Recognition of Interdatabase Dependencies", ACM SIGMOD Proc. 2. Intern. Workshop on Advances in Databases and Information Systems, Moscow, J. Eder and L.A. Kalinichenko (Eds.), Workshops in Computing Series:255-265, Springer Verlag.
- [14] Klusch, M., 1995, "Coalition-based cooperation between intelligent agents for a contextual recognition of interdatabase dependencies", Proc. 1. Intern. Conference on Multi-Agent Systems ICMA5-95, San Francisco.
- [15] Klusch, M., 1996, "Utilitarian coalition formation between information agents for a cooperative discovery of interdatabase dependencies", in: S. Kirn and G. O'Hare (Eds.), *Cooperative Knowledge Processing*, 1996, Springer Verlag, London.
- [16] Klusch, M. and Shehory, O., 1996, "Coalition formation among rational information agents", Proc. of MAAMAW-96, Eindhoven, LNAI Series Vol. 1038:204-217, Springer Verlag.
- [17] Peleg, B., 1984, *Game Theoretic Analysis of Voting in Committees*, Cambridge University Press.
- [18] Rapoport, A., 1970, *N-Person Game Theory*, University of Michigan, 1970.
- [19] Sandholm, T., Lesser, V., 1995, "Coalition formation among bounded rational agents", Proc. IJCAI-95, Montréal.
- [20] Shehory, O. and Kraus, S., 1993, "Coalition formation among autonomous agents: Strategies and complexity", Proc. of MAAMAW-93, Neuchâtel.
- [21] Shehory, O. Kraus, S., 1995, "Task allocation via coalition formation among autonomous agents", Proc. IJCAI-95, Montreal.
- [22] Shehory, O. Kraus, S., 1996, "A Kernel-oriented model for coalition formation in general environments: Implementation and results", Proc. AAAI-96, Portland.
- [23] Sheth, A., Karabatis, G., Rusinkiewicz, M., 1991 "Specifying interdatabase dependencies in a Multidatabase environment", IEEE Computer.
- [24] Stearns, R.E., "Convergent transfer schemes for n-person games", Transactions of the American Mathematical Society, 134:449-459.
- [25] Weigand, H. et al., 1995, "Integrated Semantics for Information and Communication Systems", Proc. IFIP Conference on Data Semantics .
- [26] Werner, E., 1988, "Toward a theory of communication and cooperation for multiagent planning", Proc. of the Second Conference on Theoretical Aspects of Reasoning about Knowledge, pages 129-143, Pacific Grove, CA USA.

- [27] Wooldridge, M., Jennings, N., 1995, "Intelligent Agents: theory and practice", *Knowledge Engin. Review*, 10(2), pp. 115-152.
- [28] Zlotkin, G. and Rosenschein, J.S., 1994, "Coalition, cryptography, and stability: Mechanisms for coalition formation in task oriented domains", *Proc. of AAAI-94*, pp. 432-437, Seattle, Washington USA.