Bilateral Multi-Issue Negotiation Between Active Documents and Execution Devices

Jerzy Kaczorek[1], Bogdan Wiszniewski[2]

Department of Intelligent Interactive Systems
Faculty of Electronics, Telecommunications and Informatics, Gdansk University of Technology
Gdansk, Poland
Email: [1]jkaczorek@gmail.com, [2]bowisz@eti.pg.gda.pl

***Abstract*—Mobile document-agents are often in conflict with execution devices when attempting to perform activities of the business process they implement, since preferences of device owners may change depending on their current location and the actual class of the device in use. The paper proposes a bilateral negotiation mechanism based on a simple bargaining game that can effectively resolve such conflicts without any third party support.**

Keywords–eCollaboration; Mobile computing.

I. INTRODUCTION

Individuals, who collaborate in a network organization, interact by exchanging electronic documents that constitute units of *information* and at the same time units of *interaction*. This dichotomy has become apparent with the advent of *active documents*, often implemented as software agents. In particular, a mobile interactive document (MIND) can migrate over the network and carry both, the *content* to be worked on and specification of its migration path with *activities* and *transitions* [1]. Each activity represents a piece of work to be performed by the user with the incoming document content, whereas transition indicates where the outgoing document (or documents), constituting a result of the activity, should migrate next. This idea is outlined in Figure 1; activities are represented by rectangles and transitions by arrows.

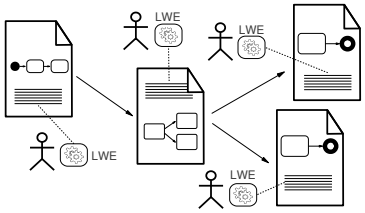

Figure 1. Mobile documents with embedded workflows

The process is started by the document originator, who sends the MIND document to the first collaborator indicated in its workflow. Each collaborator may interact with the content of the received MIND document using any currently available personal device that can receive and send email messages – from simple cellphones to smartphones or tablets to laptops or workstations. Each device has a lightweight workflow engine (LWE), implemented as an email client, which is capable of unpacking and packing the documents and sending them to other workers of the organization using email as the transport layer [1].

A. Document execution context

A single activity performed in the execution context provided by the device, on which the mobile document is currently located, depends on the policy of the document originator, operational characteristics of the currently used worker's device, and preferences of the knowledge worker responsible for the current processing step. Proactive MIND documents may handle that in several ways: activity may be performed automatically by the *embedded* document code, if allowed by the worker operating his/her device, may be performed manually by the worker using *local* services or tools installed on that device, also the device may call some *external* (third party) service requested by the document, if Internet connection is available at the time of executing the activity. This task is not trivial – as execution contexts may vary, because the same worker may use different devices when performing activities of the same business process, e.g., using a workstation when in office, a smartphone during the travel between office and home, and a laptop at home. Moreover, user preferences for the same device may depend on its current location, e.g., when out of office and accessing an untrusted network, and often conflicting with the document-agent policies. Finally, document-agents arriving to the particular device may have incomplete information on the specific execution context provided by the device.

Further in the paper, we propose *negotiation* to provide a solution to the problem of reaching an agreement between the document-agent and its execution device – even when the parties are in conflict and have incomplete information on each other preferences on how the current activity should be performed. This is the novel concept in the area of document engineering [2].

B. Negotiation model

Offers exchanged by negotiating parties are m-vectors of items $o = \langle item_1, item_2, \ldots, item_m \rangle$. Each $item_i$, where $i = 1, \ldots, m$, can be assigned a value of any attribute-specific type chosen from the set of values: $A_i = \{a_{i_1}, a_{i_2}, \ldots, a_{i_n}\}$, where $i_n = |A_i|$. Operator $|\ |$ denotes cardinality of its argument set, A_i represents the i-th attribute of the negotiated service. Set A_T of all m-vectors $A_T = \times_{i=1}^{m} A_i$ is called a *space of offers*. Based on that we define offer $o_j \in A_T$, where $j = 1, \ldots, |A_T|$, as the vector of attribute values $o_j = \langle v_1^j, v_2^j, \ldots, v_m^j \rangle$, where $v_i^j \in A_i$. Each single attribute value has an assigned numerical value, which reflects utility of the related attribute value. Utility of attribute value $a_{i_k} \in A_i$,

 ISBN: 978-1-61208-381-0

for $i_k = 1, \ldots, |A_i|$, is calculated by function $u_i : A_i \rightarrow N$. Each party has its own set of functions $\{u_1, u_2, \ldots, u_m\}$ to calculate utility of any item in the offer. Given that, utility of each offer o_j may be calculated as $U(o_j) = \sum_{i=1}^{m} u_i(v_i^j)$, where $j = 1, \ldots, |A_T|$. Throughout the rest of the paper we will use values of the utility function normalized against $\max_j(U(o_j))$, so that $U : A_T \rightarrow (0, 1]$. Formally, the problem of the paper is to find the best offer $o_c \in A_T$ that is acceptable to the document-agent P_1 and the execution device P_2, given their utility functions, U_1 and U_2. In other words o_c, called the *contract* between P_1 and P_2, maximizes their utility, i.e.,

$$o_c = \arg \max_{o \in A_T} U_1(o) U_2(o). \quad (1)$$

Since neither party knows its opponent's utility function, nor is willing to reveal its own, exchanging of offers and counter-offers is needed to systematically search space A_T. We use for that an alternating-offer protocol, modeled in the paper as the *simple bargaining game (SBG)*.

The rest of the paper is structured as follows. Section II introduces the method of representing non-functional attributes that are used to build offers exchanged by negotiating parties. The game-theoretic model of that process is defined in Section III. Next, in Section IV results of the simulation experiments are provided, which show that despite of conflicting preferences and incomplete information on each other, the document and the device may reach a satisfying solution. Section V compares the proposed method to the related proposals in the recent literature, and Section VI concludes the paper.

II. BARGAINING SETS

We have implemented the generic bilateral negotiation model introduced above for $m = 5$ attributes, thus $A_T = \times_{i=1}^{5} A_i$, $i = 1, .., 5$. Each set A_i contains attribute values that specify respective options of the execution context.

1) Performer of the activity (A_1): Disjoint subsets $D, W, J \subset A_1$ contain values indicating respective contexts where the document is allowed to execute automatically on the device (D), where it is not able or is not allowed to execute on its own, so only the user (worker) can perform the activity on its content (W), or where the activity is supposed to be executed jointly by the document and the worker (J).

2) Availability of network resources (A_2): Disjoint subsets $S, E, I \subset A_2$ contain values indicating respective contexts where the execution device is actually separated from its host organization, i.e., no network connection exists or is not allowed by the device owner (S), is connected from outside of its host organization (E), or alternatively from inside of it (I).

3) Performance of network resources (A_3): Disjoint subsets $U, R, M, A, N \subset A_3$ contain values indicating respective contexts where the parameters of the network connection (if any) are unknown (U), or optionaly, wireless (R), telephone modem (M), asymmetric digital subscriber line (ADSL) modem (A), or twisted pair (wire) connection (N), is used by the execution device.

4) Security of the network connection (A_4): Disjoint subsets $P, K, T, C \subset A_4$ contain values indicating respective contexts where the connection (if any) is not secure at all (P), uses wireless (if any) protected by the access key (K), can connect to remote sites using the secure transfer protocol (T), or combines the latter two mechanisms to provide the most secure connection possible (C).

5) Reliability mechanisms supporting interaction (A_5): Disjoint subsets $L, B, F, H \subset A_5$ contain values indicating respective contexts where no support is provided by the document embedded functionality or the execution device system to protect the document content from user errors (L), some back-up support is provided by the autosave option (B), failsafe option is provided by the acceptance button, i.e., no changes to the content are permanent until accepted by the worker (F), or high reliability can be provided by combining the later two with the "undo" button and the automatic check of the content performed by the document itself (H).

Note that based on the above model each single vector (offer) $o \in A_T$ specifies in fact a concrete, multi-aspect execution context that may be negotiated by the MIND document and its currently available execution device.

A. Multi-option offers

In our model, the space of offers A_T is discrete, as values of the respective attributes constituting each offer are selected from enumerable sets of available options. Tables I–V illustrate our approach to modeling of multi-option (multi-issue) offers. Specific options that contribute to each respective attribute value $v_i^j \in A_i$, are represented by binary flags. The label of each respective option considered for the given attribute value is listed in the header of each corresponding table, values '0' and '1' listed in each respective column below the option label indicate options 'is' or 'is not' present when calculating value of the given attribute; if the flag may assume two possible values in the context defined by the given table row we denote that by the regular expression $[0, 1]$. The respective attribute values are generated by combining flag values in each row, which we also specify with regular expressions, listed in the rightmost column of each corresponding table. The first letter denotes the respective subsets of A_i, explained in p.II-1–II-5, while digits indicate each meaningful combination of option flags, considered in our current implementation of MIND.

Option values for attribute A_1 are listed in Table I and specify potential performers required by the document to complete its activity: the *Worker* (*Wkr*) using it, the *Embedded Service* (*EmS*) brought by it to the device, some *External Service* (*ExS*) the device should allow it to call, any *Local Tool* (*LoT*) the document may want to use, and any *Local Service* (*LoS*) the document may want to access, when interacting with the local operating system of the device. Combinations of these options' flags distinguish in total nine different subsets of A_1, labeled with symbols listed in the rightmost column of Table I.

TABLE I. EXECUTION CONTEXT OPTIONS FOR 'PERFORMER'

Wkr	*EmS*	*ExS*	*LoT*	*LoS*	*Option labels*
0	1	[0,1]	0	[0,1]	*D*[1-3]
1	0	[0,1]	1	[0,1]	*W*[1-2]
1	1	[0,1]	[0,1]	[0,1]	*J*[1-4]

For example, a proactive document that intends to perform its activity without interacting with the worker using the execution device ($Wkr = 0$), only by the means of its embedded functionality ($EmS = 1$) and some external service ($ExT = 1$), would not need any local tool ($LoT = 0$) or

service ($LoS = 0$) installed on the device; this particular 'performer' attribute value '01100' would be labeled with $D3$ in our model.

Option values for attribute A_2 are listed in Table II; they specify various resources that should be available to the document when performing its activity on the device. The device connected from inside of the worker's organization has a *Local IP* (*LIP*), or otherwise an *External IP*. The *Specific Browser* (*SpB*) required by the document may be available on the device, or just *Any Browser* (*AnB*). Similarly, the *Specific Tool* (*SpT*) requested by the document may be provided by the local operating system, or just any *Substitute Tool* (*SuT*). Moreover, the device may be equipped with the *Full Keyboard* (*FKb*), or alternatively a smaller set of *Selection Buttons* (*SeB*) can be provided. Combinations of these options' flags distinguish in total 22 different subsets of A_2, labeled with symbols listed in the rightmost column of Table II.

TABLE II. EXECUTION CONTEXT OPTIONS FOR 'AVAILABILITY'

LIP	*EIP*	*SpB*	*AnB*	*SpT*	*SuT*	*FKb*	*SeB*	*Option labels*
0	0	0	0	[0,1]	[0,1]	[0,1]	[0,1]	S[1-8]
0	1	[0,1]	[0,1]	[0,1]	[0,1]	[0,1]	[0,1]	E[1-7]
1	0	[0,1]	[0,1]	[0,1]	[0,1]	[0,1]	[0,1]	I[1-7]

For example, a proactive document that is ready to perform its activity on the device connected from outside of its host organization ($LIP = 0$ and $EIP = 1$) may not care about the type of browser ($SpB = [0, 1]$ and $AnB = [0, 1]$), and assumes no other support from the local system ($SpT = SuT = 0$) nor the device ($FKb = SeB = 0$); four possible 'availability' attribute values '01**0000' would be considered equal and labeled with $E1$ in our model.

Option values for attribute A_3 are listed in Table III and specify performance aspects of the execution device during the current activity. The connection may be *Wired* (*Wre*), or using a TV *Cable* or telephone *Line* (*C/L*), the plain *Telephone Modem* (*TMo*), or any *Wireless* (*Wrs*) network within the reach of the device. The device's processor (*CPU*) may be above the average specified by the document, as well as the device may provide more memory (*RAM*) as the minimum required by the document. Combinations of these options' flags distinguish in total 20 different subsets of A_3, labeled with symbols listed in the rightmost column of Table III.

TABLE III. EXECUTION CONTEXT OPTIONS FOR 'PERFORMANCE'

Wre	*C/L*	*TMo*	*Wrs*	*CPU*	*RAM*	*Option labels*
0	0	0	0	[0,1]	[0,1]	U[1-4]
0	0	0	1	[0,1]	[0,1]	R[1-4]
0	0	1	0	[0,1]	[0,1]	M[1-4]
0	1	0	0	[0,1]	[0,1]	A[1-4]
1	0	0	0	[0,1]	[0,1]	N[1-4]

For example, a proactive document accessing only a wireless network ($Wre = C/L = TMo = 0$ and $Wrs = 1$) during its current activity, accepting less powerful CPU ($CPU = 0$) but consuming RAM above average ($RAM = 1$) would have its 'performance' attribute value '000101' labeled with $R2$ in our model.

Option values for attribute A_4 are listed in Table IV and specify security aspects of the current activity. Data can be *Securely Transferred* (*SeT*) if the remote site provides HTTPS. Moreover, if the device uses wireless connection its security may be improved if the network is protected by the *Access Key* (*AcK*). The document may also be *Digitally Signed* (*DSg*), and the local system of the device may be protected by some *Anti-Virus* (*AnV*) tool. Combinations of these options' flags distinguish in total 16 different subsets of A_4, labeled with symbols listed in the rightmost column of Table IV.

TABLE IV. EXECUTION CONTEXT OPTIONS FOR 'SECURITY'

SeT	*AcK*	*DSg*	*AnV*	*Option labels*
0	0	[0,1]	[0,1]	P[1-4]
0	1	[0,1]	[0,1]	K[1-4]
1	0	[0,1]	[0,1]	T[1-4]
1	1	[0,1]	[0,1]	C[1-4]

For example, a proactive document not requiring a secure transfer for its data ($SeT = 0$) but expecting the wireless network protected by the access key ($AcK = 1$), with its content digitally signed ($DSg = 1$) and the local system protected with some antivirus software ($AnV = 1$) would have its 'security' attribute value '0111' labeled with $K4$ in our model.

Option values for attribute A_5 are listed in Table V and specify reliability aspects of the operations performed on the document content during the activity. If the *Acceptance Button* (*AcB*) is provided the user can decide on permanence of the document content modifications. The *Autosave Mode* (*ASv*) provided by the related tool or service can prevent the user from loosing accidentally the content entered so far. Functionality providing any *Automatic Check* (*ACh*) of the content being entered by the worker may improve its correctness, whereas the *Undo Button* (*UdB*) would improve comfort of work of the worker and further reduce the rate of errors he/she can make when modifying the document content. Combinations of these options' flags distinguish in total 16 different subsets of A_5, labeled with symbols listed in the rightmost column of Table V.

TABLE V. EXECUTION CONTEXT OPTIONS FOR 'RELIABILITY'

AcB	*ASv*	*ACh*	*UdB*	*Option labels*
0	0	[0,1]	[0,1]	L[1-4]
0	1	[0,1]	[0,1]	B[1-4]
1	0	[0,1]	[0,1]	F[1-4]
1	1	[0,1]	[0,1]	H[1-4]

For example, if the proactive document performs the activity entirely on its own, it may reasonably expect the device to provide just the acceptance button ($AcB = 1$ and $ASv = ACh = UdB = 0$), to allow the worker to accept the concluded activity and send it to the next activity of its workflow. The related 'reliability' attribute value '1000' would be labeled with $F1$ in our model.

B. Bargaining over option trees

Realistically, negotiating parties will continue choosing offers from a certain subset of offers $C_B \subset A_T$, which we call the *bargaining set*. The range of options, available to the particular document-agent in the given execution context, is determined by the type of a personal device currently in use by the worker. In other words, the device class defines a concrete bargaining set content. We distinguish five basic classes of devices in the current implementation of MIND.

1) Workstations: They are immobile and used mainly at the user's workplace (in office) or at home. If used in office they are usually wired from inside to the organization's network, have access to various secure services, offer reliable interfaces and have relatively high computational power. If used at home they offer the similar level of service, except for network connections that may be external to the organization – if no virtual private network (VPN) connection is possible – and may use various types of modems.

2) Laptops: Performance of their hardware makes them not less powerful than workstations, but owing to their ability to access networks in many ways, including wire, WiFi, and modems, they are more versatile. The only distinction that may be taken into account when characterizing execution contexts they provide is the software they use. If the laptop is a private property of the worker, it may lack some proprietary software tools provided by the organization to its workers, so sometimes substitute tools may have to be used to perform specific activities on the document content – especially when performed from outside of the organization.

3) Tablets: They usually have less computational power than laptops, and are less versatile, due to the limited range of networking solutions they support (embedded WiFi cards or/and ADSL modems). Some specialized software (local tools or services) required to handle properly the content of the document-agent may be unavailable, what can affect results of the activity to be performed.

4) Smartphones: Although their recent technological advances are impressive, they may lack (like tablets) specific local tools or services required to properly process the content brought to the device by the document-agent. These devices also slightly differ from tablets in the networking solutions they support – since the telephone modem may be used as the alternative to WiFi. Reliability of interaction is usually slightly reduced compared to tablets, due to their smaller screen sizes.

5) Cellphones: They are the weakest execution devices, although they can support most elementary execution scenarios performed by workers on the document content, e.g., simple form filling or modifying/accepting simple text prepared by someone else. These operations can be performed if the cellphone currently in use by the worker is able to read email messages, usually via the cellular network.

In Table VI, we specify bargaining sets for various classes of devices listed above. The range of option values of the five attributes in our current MIND implementation reflects the specificity of each device characterized above, which is connected to some network and willing to use it during the entire activity to be performed.

TABLE VI. BARGAINING SETS FOR EXECUTION DEVICES

Bargaining set:	*Attribute A_i value sets/subsets*				
device	1	2	3	4	5
C_1: workstation	A_1	$E \cup I$	$M \cup A \cup N$	$P \cup T$	A_5
C_2: laptop	A_1	$E \cup I$	$A_3 - U$	A_4	A_5
C_3: tablet	A_1	E[4-6]$\cup I$[4-6]	$R \cup A$	A_4	A_5
C_4: smartphone	A_1	$\{E3, I3\}$	$R \cup M$	A_4	L
C_5: cellphone	A_1	E[1-2]	M[1-2]	$P \cup T$	$\{L1\}$

Notice that the range of offers which may be exchanged during negotiation between the document-agent and the device of any particular class is limited, i.e., $C_1, .., C_5$ are rather small subsets of A_T. Either negotiating party, the document and the device, selects offers from the corresponding set C_i independently, with regard to its own valuation of each single offer. The alternating-offer protocol assumed by MIND requires parties to individually sort all offers from the relevant bargaining set in the order implied by that valuation. The ordering of offers preferred by each party is represented by the related *option tree*.

Consider two parties, a document-agent and a laptop, negotiating over C_2; let it consist of just five offers:

$$\begin{aligned} o_1 &= < D3, E1, A4, T4, H3 >, \\ o_2 &= < D3, E1, M4, T4, H3 >, \\ o_3 &= < D3, E1, R2, K4, F1 >, \\ o_4 &= < D3, E1, R4, C4, H1 >, \\ o_5 &= < D3, I1, N4, T4, H4 > . \end{aligned}$$

Detailed interpretation of the example offer o_3 attribute values has been presented before in connection to Tables I-V, while the valuation and preferences of all options negotiated by the document and the device are shown in Figures 2 and 3.

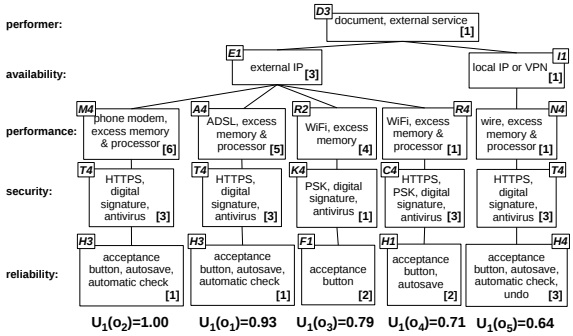

Figure 2. Example option tree of the proactive document

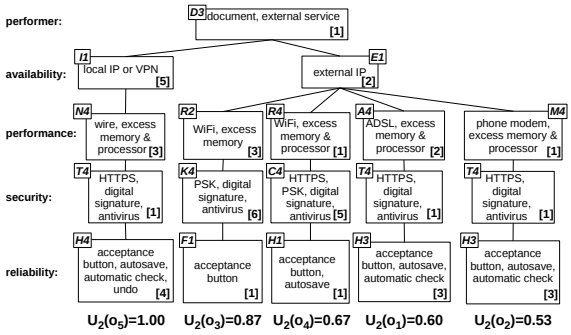

Figure 3. Example option tree of the laptop execution device

Nodes in each example option tree are labeled with option values specified in Tables I–V, while utility of each single attribute value calculated by the party is specified in square brackets. Utilities of the respective offers of C_2 are listed at the bottom of the tree in a normalized form, as explained before; see in Figure 2 that o_2 is the document's most preferred offer, since $\sum_{i=1}^{5} u_i(v_i^2) = 1 + 3 + 6 + 3 + 1 = 14$ and $U_1(o_2) = 14/14 = 1.0$, whereas o_5 is its least preferred offer, with $U_1(o_5) = 8/14 = 0.64$.

III. SIMPLE BARGAINING GAME

We model negotiation between the document-agent and the execution device as a *multi-stage game*. It consists of *stages*,

denoted as $(\alpha_1^{2n}, \alpha_2^{2n+1})$, where $n \in N \cup \{0\}$ (or $n \in Z_{\geq 0}$ for brevity) denotes the stage number, α_i^k denotes a *move* of each respective player P_i, $i = 1, 2$, and k is the move number. After making move α_i^k player P_i chooses offer $o \in C_B$ and valuates it with its payoff function $\pi_i : A \times N \rightarrow [0, 1]$, calculated as $\pi_i(o, k+2) = \delta_i \pi_i(o, k)$, where $\delta_i \in (0, 1]$ is called a *discount factor* and $\pi_i(o, 0) = U_i(0)$. Each player has its own discount factor that remains constant during the game. Throughout the rest of the paper we will denote the opponent of player P_i by P_{-i}, their respective utility functions by U_i and U_{-i}, payoff functions by π_i and π_{-i}, and discount factors by δ_i and δ_{-i}.

Rules of our simple bargaining game are the following:

1) The game is started by player P_1.
2) Players P_i, $i = 1, 2$ keep in secret their private information, including U_i, δ_i and π_i, but share knowledge on the bargaining set C_B.
3) Utility values of players' offers are discounted at each transition to the next stage.
4) Players exchange offers until the game is concluded, i.e., one of the players accepts an offer or quits the game.
5) The game is concluded by player P_i when:
   a) P_i repeats its own offer o' what implies quitting the game by P_i.
   b) P_i repeats player's P_{-i} offer o'' what implies accepting it as the contract and exiting.

A. Negotiation algorithm

A generic form of the algorithm implemented by each SBG player (thread) is given below. Two threads operate on the same resource, which is the bargaining set and individual indexes for marking offers in it as sent, received or not yet sent nor received. Functions of the form $max\Pi_{[B,R,N]}$ calculate offers to be chosen from the bargaining set or its respective subsets, according to π_i and the move number k, as explained before.

Public: Bargaining set C_B; sets of offers: received C_R, sent C_S, remaining C_N (all initially empty); received o_R and sent o_S offers; move number k (initially $k = 0$).
Private: Discount factor δ_i; payoff function π_i;

if $k = 0$ **then**
  {Opening move}
  $o_S \leftarrow max\Pi_B$;
  $send(o_S)$;
else
  $receive(o_R)$;
  if $o_R \in C_S$ **then**
    {Opponent has accepted the offer}
    $terminate$;
  end if
  $C_R \leftarrow C_R + o_R$;
  $C_N \leftarrow C_B - (C_R + C_S)$;
  if $C_N = \emptyset$ **then**
    {Last move}
    $o_S \leftarrow max\Pi_R$;
    $send(o_S)$; $terminate$;
  else
    {Intermediate/penultimate move}
    $o'_S = max\Pi_R(k)$; $o''_S = max\Pi_N(k)$;
    if $\pi_i(o'_S) \geq \delta_i \cdot \pi_i(o''_S)$ **then**
      $send(o'_S)$;
    else
      $send(o''_S)$;
    end if
  end if
  $k = k + 1$
end if

B. Collaboration agreement

Table VII specifies history of SBG played by document-agent P_1 and execution device P_2, which shared the example bargaining set $C_2 = \{o_i | i = 1, \ldots, 5\}$, with their respective option trees specified in Figures 2 and 3. The document started negotiation by offering to the device its most preferred option o_2 in move α_1^0. Then after the next four moves, option o_3 offered by the device was finally accepted by the document. The negotiated contract between the proactive document and the device provides the former with the commonly agreed *execution context* supplied by the latter.

TABLE VII. BARGAINING OVER EXAMPLE OPTION TREES

Stage	Move	Player	Offer o_i	$U_i(o_i)$	$U_{-i}(o_i)$
0	0	P_1	o_2	1.00	0.53
0	1	P_2	o_5	1.00	0.64
1	2	P_1	o_1	0.93	0.60
1	3	P_2	o_3	0.87	0.79
2	4	P_1	o_3	0.79	agreed

In the example we have used simplified option trees to keep them small. In real applications involving MIND documents the upper bound for the maximum size of a single option tree could be as high as the product of the numbers of attribute value labels used in Tables I-V, which is well over 5000, whereas the upper bound for the number of possible negotiation histories is a product of their respective permutations, in the order of magnitude of 10^{13}.

IV. SIMULATION EXPERIMENT

Our simple bargaining game can provide a solution of (1) when neither player P_i knows its opponent's U_{-i}, δ_{-i} nor π_{-i}, known in the literature as the *Nash equilibrium*.

In order to show the above let us formally define the simple bargaining game $SBG = \{P, D, U, C_B, S, \Pi, T\}$ as a multi-stage game. Sets $P = \{P_1, P_2\}$, $D = \{\delta_1, \delta_2\}$ and $U = \{U_1, U_2\}$ are self explanatory. In the bargaining set C_B we distinguish subsets $C_i, C_{-i} \subset C_B$ of offers, submitted respectively by P_i and P_{-i}. Therefore, the set including offers not yet submitted by any party would be $C' = C_B - (C_i \cup C_{-i})$, and $C' \subset C_B$. Set $S = S_i \cup S_{-i}$ consists of strategies used by the respective players; each single strategy $s_i \in S_i$ is a function $s_i : Z_{\geq 0} \rightarrow C_B$ that associates each possible move α_i^{k+2} of P_i that follow α_i^k with the relevant offer from C_B. There are at most $|C_B|$ possible strategies for each player P_i, while the maximum number of steps is $k_{max} = |C_B|$. Set $\Pi = \{\pi_1, \pi_2\}$ consists of the respective players' payoff functions; for any offer $o \in C_B$, submitted in the k-th move, each respective payoff function returns $\pi_i(o, k) = \delta_i^k U_i(o)$ if $o \in C_{-i}$, or $\pi_i(o, k) = 0$ if $o \in C_i$. Finally, $T(o)$ denotes a condition for concluding the game, i.e., for any $o \in C_B$, $T(o) = false$ if $o \notin (C_i \cup C_{-i})$, otherwise $T(o) = true$.

A. Equilibrium strategy

For each stage of SBG involving offer o_i, submitted by P_i, and counteroffer o_{-i}, submitted by P_{-i}, the notion of a *strategy profile* is used; it is defined as a pair of strategies $\langle s_i, s_{-i}\rangle$, where $s_i \in S_i$ and $s_{-i} \in S_{-i}$. We will say that the strategy profile $\langle s'_i, s'_{-i}\rangle$ is the Nash equilibrium of SBG if $\langle \pi_i(s'(k)), \pi_{-i}(s'_{-i}(k+1))\rangle \succeq \langle \pi_i(s(k)), \pi_{-i}(s'_{-i}(k+1))\rangle$ for any $s_i \in S_i$ and each player in each stage n; by '$\succeq$' we denote a pairwise comparison operator of 2-element vectors. Strategies s'_i and s'_{-i} are called *equilibrium strategies*; each one is said to provide the *best offer* of the respective player in each stage of the game. When looking for the solution to (1), instead of attempting to find each other's equilibrium strategies each player may just assume that the offers submitted by its opponent in any stage are actually its best responses.

When implementing the simulation experiment we have adopted the following rationale of strategies used by players in various stages of the game. All estimations are made from the point of view of P_i, thus if the move is to be made by P_i, we assume it to choose the offer of the highest payoff from all offers available to it. Alternatively, if the move is to be made by P_{-i}, we assume it to choose the offer of the average payoff calculated for all offers available to it. The reason for the latter is that from the perspective of P_i all offers made by P_{-i} are equally probable, so statistically the payoff that P_i can get is equal to their average value. We calculate that average with the auxiliary function $\rho(o, C', f) = 1/|C'| \sum_{o \in C} f(o)$, where $o \in C'$, and $f : C' \to [0, 1)$ is a function used to valuate offers with the payoff function. For brevity we will skip using the step number argument of the payoff function throughout the rest of the paper.

1) Last move: Strategy $s_i(k_{max}) = \arg\max_{o \in C_{-i}} \pi_i(o)$ is used if P_i makes the last move. Alternatively, if the last move is made by P_{-i}, strategy $s_{-i}(k) = o_{-i}$ is used, where $o_{-i} \in C_i$ satisfies equation $\pi_i(o) = \rho(o, C_i, \pi_i)$. In the first case P_i must choose the best offer from those that have been already presented by P_{-i}, for otherwise (according to rules of *SBG*) it would get the zero payoff. For the same reason, offer o_{-i} made by P_{-i} must be chosen from all available offers in C_i and of the payoff closest to the average payoff of all its offers.

2) Penultimate move: The strategy that has to be used is $s_i(k) = \arg\max_{o \in \{o'_i, o''_i\}}(\pi_i(o'_i), \delta_i \pi_i(o''_i))$, if P_i makes the move, where $o'_i = \arg\max_{o \in C_{-i}} \pi_i(o)$ and $o''_i \in C - C_{-i}$ satisfies $\pi_i(o) = \rho(o, C_i, \pi_i)$. Alternatively, $s_{-i}(k) = o_{-i}$ is used if P_{-i} makes the move, where $o_{-i} \in C - C_{-i}$ satisfies $\pi_i(o) = 0.5(\pi_i(o'_{-i}) + \delta_{-i}\pi_i(o''_{-i}))$, $o'_{-i} \in C_i$ satisfies $\pi_i(o) = \rho(o, C_i, \pi_i)$, and $o''_{-i} = \arg\max_{o \in C_B - C_i} \pi_i(o)$. Unlike in the last move, the player has to accept one of the received offers or make a new one, not yet presented by either player. In the penultimate move only one such offer is left in C_B. If P_i makes the move it finds o'_i giving the best payoff of all offers received already from P_{-i} (as described in p. IV-A1 before) and estimates payoff of offer o''_i that P_{-i} may submit in its next move. The latter may be any offer submitted already by P_i and the last one left in C_B – all equally probable, thus the average payoff is assumed. The choice is between o'_i and discounted o''_i, since the latter concerns continuing the game in the next move. If P_{-i} makes the penultimate move it can either accept offer o'_{-i} from those made already by P_i or reject it by submitting counteroffer o''_{-i}. In order to assess payoff of o'_{-i} player P_i has to calculate the average payoff of all its offers to P_{-i} (in the case when P_{-i} may decide to accept one of them), whereas o''_{-i} is expected to give the highest payoff of the offers made by P_{-i} that P_i may eventually accept. For o'_{-i} the average payoff is calculated, as accepting P_i's offers by P_{-i} is equally probable. The choice is then between o'_{-i} and discounted o''_{-i}, since the latter concerns continuing the game in the next move. The average payoff of o'_{-i} and o''_{-i} is assumed, as both choices by P_{-i} are equally likely to P_i.

3) Intermediate move: In any move $0 < k < k_{max} - 1$ P_i uses strategy $s_i(k) = \arg\max_{o \in \{o'_i, o''_i\}}(\pi_i(o'_i), \delta_i \pi_i(o''_i))$, where $o'_i = \arg\max_{o \in C_{-i}} \pi_i(o)$, and $o''_i \in C'$ satisfies $\pi_i(o) = \max_{o \in C'}(1 - \delta_{-i})\pi_i(o) + \delta_{-i} U_i(s_{-i}(k+1))$. Alternatively, strategy of P_{-i} should be $s_{-i}(k) = o_{-i}$, where $o_{-i} \in C'$ satisfies $\pi_i(o) = 0.5(\pi_i(o'_{-i}) + \delta_{-i}\pi_i(o''_{-i}))$, $o'_{-i} \in C_i$ satisfies $\pi_i(o) = \rho(o, C_i, \pi_i)$, and $o''_{-i} \in C'$ satisfies $\pi_i(o) = (1 - \delta_i)\rho(o, C') + \delta_i \pi_i(s_i(k+1)))$. Rationale for strategies $s_i(k)$ and $s_{-i}(k)$ is the same as in the penultimate move, i.e., offers o'_i and o'_{-i} concern accepting the respective opponent's offer, while o''_i and o''_{-i} concern rejection of the opponent's offer and continuation of the game by submitting new offers. The former are calculated in the same way as in p. IV-A2, whereas calculation of the latter should reflect possible acceptance or rejection during subsequent stages. Therefore, if P_i decides to continue the game the estimated payoff splits in two parts, proportional to discount factor δ_{-i} of its opponent: payoffs that may be get for new offers from C' if accepted, and continuation of the game by player P_{-i} in the next move with strategy $s_{-i}(k+1))$ if not accepted. If P_{-i} decides to continue the game the estimated payoff splits in two parts as well, proportional to discount factor δ_i of its opponent: the average payoff that may be get for new offers from C', if accepted, and continuation of the game by player P_i in the next move with strategy $s_i(k+1)$, if not accepted.

4) Opening move: Strategy of P_1 in move $k = 0$ (starting the game) is $s_i(0) = \arg\max_{o \in C_B}(U_i(o))$. It has no hints concerning its opponent's utility, so the only way to maximize its utility is to choose its most valued offer from C_B.

B. Simulation results

In the experiment we simulated P_1 (proactive documents) using the rationale of selecting best offers in each possible type of step described before. Some generic features of such documents were considered. We distinguished *protected* documents from *open* ones, by taking into account whether they need any secure connection to perform their activity or not, and *heavy* documents from *light* ones, by considering whether they may require CPU power and the amount of RAM above or below some average levels when doing that. Based on that we have defined four classes of documents used in the simulation experiments. Their respective bargaining sets are listed in Table VIII.

TABLE VIII. BARGAINING SETS FOR PROACTIVE DOCUMENTS

Bargaining set:	*Attribute A_i value sets/subsets*				
document class	1	2	3	4	5
C_{ph}: protected & heavy	A_1	$E \cup I$	$[U, R, A, N]4$	$A_4 - P$	A_5
C_{pl} : protected & light	A_1	$E \cup I$	A_3	$A_4 - P$	A_5
C_{oh}: open & heavy	A_1	$E \cup I$	$[U, R, A, N]4$	A_4	A_5
C_{ol} : open& light	A_1	$E \cup I$	A_3	A_4	A_5

For each document class 20 random option trees were generated, each one including offers with attribute values listed

in the respective row of Table VIII. Bargaining was performed over each possible pair of options trees, one tree for the particular document class and another for the particular device class; in each case $C_B = C_{doc} \cap C_{dev}$. With five device classes 100 simulations were performed for each document class. In each single simulation the offers in option trees of the respective negotiating parties were sorted in the opposite order to each other, to ensure the maximum possible number of negotiation steps. The discount factor for each party was $\delta = 0.8$. The results are shown in Figure 4.

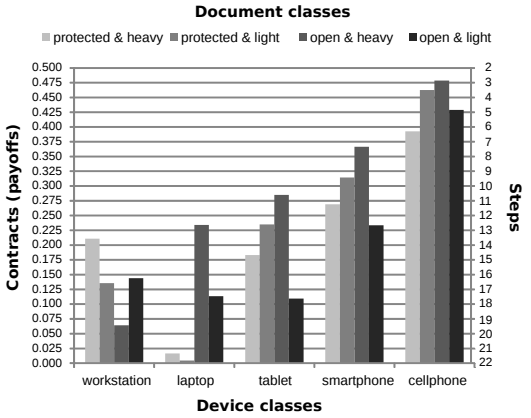

Figure 4. Contracts negotiated by proactive documents and devices

The fair payoff, which may be calculated based on (1) by some hypothetical third party knowing utility functions of the two negotiating parties, was close to 0.5. It may be seen that in general the smaller the number of available options (thus also the bargaining set size) the smaller is the number of negotiation steps required to reach the contract – the document was able to reach the agreement and get close to the fair result with the cellphone device in just a few steps, whereas negotiating with the laptop or workstation devices was possible in about 20 steps.

V. Related work

Our problem of finding the execution context that can satisfy the proactive document intending to perform a given activity on a dynamically changing mobile device, relates to two specific areas: representation of available options as offers, and implementing negotiations as games. One difficulty with the above is the discrete range of offers that the negotiating parties have to search to find the solution. Another is the semi-cooperative setting implied by the fact that partners share the bargaining set but keep their preferences secret. Modeling multiple options with functions as the continuum of offers allows finding contracts using various optimization techniques, e.g., swarm optimization, as demonstrated in [3]. Unfortunately, dealing with non-functional options makes such optimization inherently difficult to implement. In [4] the concept of the Web service modeling ontology (WSMO) has been used to develop a mechanism capable of handling that in a way enabling agents to find the most suitable resources for performing the requested activity; the proposed mechanism also provided for resolving between alternative (conflicting) offers based on the argumentation theory. This approach suits well broker agents recommending services to user agents, however in the case of two agents competing to win as much wealth as possible the game theoretic approach seems to be easier to implement.

A generic approach to the problem of multiple issue negotiation with no information about the opponent has been proposed in the literature [5], but the formal mathematical proof of the convergence of the monotonic concession strategy, which our SBG implements with option trees, was not provided until [6]. It has been shown there that offers and counteroffers, selected by each party according to the amount of concession the party can accept in the current round, are getting closer in the utility space until the contract is agreed. For each party a hyperquadratic utility function was assumed – general enough to simulate negotiation protocols, but imposing an unnecessary limit on how agents may implement valuation of offers and not allowing for discrete values.

VI. Conclusion

Our approach based on option trees does not assume any class of utility functions, except that they should be injective in order to enable sorting the trees and to ensure monotonicity of preferences. The linear additive utility function defined in Section I-B has been used by us just to simplify generation of option trees for simulation experiments described in Section IV-B. The model proposed in the paper has been recently expanded with the learning capability, based on the history of interaction and the concept of policies. Simulation experiments indicate that properly trained document-agents can recognize preferences of devices and guess the contract in just a couple of steps [7]. The necessary 'knowledge' is carried by documents as weights needed to set up a really simple neural network, which the LWE client would be able provide on the execution device.

Acknowledgment

This work was supported in part by the National Science Center, Poland, under grant DEC1-2011/01/B/ST6/06500.

References

[1] M. Godlewska and B. Wiszniewski, "Smart email: Almost an agent platform," in Innovations and Advances in Computing, Informatics, Systems Sciences, Networking and Engineering, ser. Lecture Notes in Electrical Engineering, T. Sobh and K. Elleithy, Eds. Springer International Publishing, 2015, vol. 313, pp. 581–589.

[2] J. Kaczorek and B. Wiszniewski, "Augmenting digital documents with negotiation capability," in Proc. 2013 ACM Symposium on Document Engineering (DocEng'13). ACM, Sep. 2013, pp. 95–98, doi:10.1145/2494266.2494305.

[3] Z. Wang and L. Wang, "Adaptive negotiation agent for facilitating bi-directional energy trading between smart building and utility grid," IEEE Transactions on Smart Grid, vol. 4, Mar. 2013, pp. 702–710, doi:10.1109/TSG.2013.2237794.

[4] A. Caballero, A. Muòoz, J. Soto, and J. A. Botía, "Resource assignment in intelligent environments based on similarity, trust and reputation," J. Ambient Intell. Smart Environ., vol. 6, Mar. 2014, pp. 199–214, doi:10.3233/AIS-140253.

[5] G. Lai and K. Sycara, "A generic framework for automated multi-attribute negotiation," Group Decision and Negotiation, vol. 18, Jul. 2009, pp. 169–187, doi:10.1007/s10726-008-9119-9.

[6] R. Zheng, N. Chakraborty, T. Dai, K. Sycara, and M. Lewis, "Automated bilateral multiple-issue negotiation with no information about opponent," in Proc. 46^{th} Hawaii Int. Conf. on System Sciences (HICSS 2013). IEEE, Jan. 2013, pp. 520–527, ISBN: 978-0-7695-4892-0.

[7] J. Kaczorek and B. Wiszniewski, "Document agents with the intelligent negotiation capability," in Proc. Knowledge and Cognitive Science and Technologies (KCST 2015). IIIS, Jul. 2015, in press.