Approximate Reasoning Based on Approximation Spaces

Andrzej Skowron[1], Jarosław Stepaniuk[2], James Peters[3], and Roman Swiniarski[4,5]

[1] Institute of Mathematics, Warsaw University
Banacha 2, 02-097 Warsaw, Poland
skowron@mimuw.edu.pl
[2] Department of Computer Science, Białystok University of Technology
Wiejska 45a, 15-351 Białystok, Poland
jstepan@ii.pb.bialystok.pl
[3] Department of Electrical and Computer Engineering, University of Manitoba
Winnipeg, Manitoba R3T 5V6, Canada
jfpeters@ee.umanitoba.ca
[4] Institute of Computer Science, Polish Academy of Sciences
Ordona 21, 01-237 Warsaw, Poland
[5] Department of Mathematical and Computer Sciences, San Diego State University
5500 Campanile Drive San Diego, CA 92182, USA
rswiniar@sciences.sdsu.edu

Abstract. This paper considers the problem of approximate reasoning in the context of approximation spaces. Zdzisław Pawlak introduced Approximation spaces in his seminal work on rough sets more than two decades ago. In this paper, we show that approximation spaces are basic structures for machine learning and pattern recognition. The utility of approximation spaces as fundamental objects constructed for concept approximation is emphasized. Examples of basic concepts are given throughout this paper to illustrate how approximation spaces can be beneficially used in many settings. The contribution of this paper is the presentation of an approximation space-based framework for doing research in various forms of learning, especially reinforcement learning in non-stationary environments as well as hierarchical learning in distributed environments.

Keywords: rough sets, approximation spaces, concept approximation, distributed environment, learning, neighborhood, rough inclusion, uncertainty function.

1 Introduction

Approximation spaces are fundamental structures for the rough set approach [17, 18, 39]. In this paper we present a generalization of the original approximation space model. Using such approximation spaces we show how the rough set approach can be used for approximation of concepts assuming that only partial information on approximation spaces is available. Hence, searching for concept approximation, i.e., the basic task in machine learning and pattern recognition can be formulated as searching for relevant approximation spaces. In the paper we also characterize the operations on approximation spaces, called constrained sums, that are used in searching for complex concept approximation. We also discuss an important role of constrained sums in hierarchical modelling and in approximate reasoning. The constrained sums are generic operations for approximate reasoning in distributed environments and in multiagent systems.

Our approach is also related to the perception based computing. Studying cognition and in particular, perception based computing [1, 2, 9, 10, 8, 13–15, 46] is becoming now one of the very active research direction for methods of complex concept approximation [5, 4, 16, 23, 19, 48] and in the consequence for building intelligent systems.

The paper is organized as follows. In Section 2 we recall the definition of approximation spaces. Next, we describe a constructive approach for computing values of uncertainty and rough inclusion functions. These functions are the basic components of approximation spaces. Parameters of the uncertainty and rough inclusion functions are tuned in searching for relevant approximation spaces. We distinguish among such parameters sensory environments and their extensions. These parameters are used for constrictive definition of uncertainty and rough inclusion functions. In particular, we show how information systems can be defined from such approximation spaces. We recall the basic definitions of concept approximation using the approximation spaces. The presented examples of approximation spaces are showing that the discussed approach generalizes several known approaches to approximation in rough set theory. Next, in Section 3 we discuss the problem of concept approximation under assumption that only a partial information about approximation spaces is available. In this case to decide if a given object belongs to the upper or lower approximation of a given concept it is necessary to estimate the exact value of the rough inclusion function for the neighborhood of this object and the approximated concept because the exact value may be not available. We present an illustrative example for such estimation following a well known heuristic used for rule based classifier construction. In this way, we are showing that searching for relevant approximation spaces is closely related to the basic task of classifier construction in machine learning and in pattern recognition. Searching methods for relevant approximation spaces for complex concept approximation are important in hierarchical learning and in approximate reasoning, in particular in distributed environments and multiagent systems. The search space is equal to the set generated from some generic approximation spaces and some special operations on approximation spaces that we call constrained sums. We discuss applications of such spaces for complex concept approximation using hierarchical learning, approximation of vague dependencies and ontology approximation. Section 4 considers operations on approximation spaces, and Section 5 describes an approach to reinforcement learning using approximation spaces.

2 Approximation Spaces

In this section we recall the definition of an approximation space from [32, 42].

Definition 1. *A parameterized approximation space is a system* $AS_{\#,\$} = (U, I_{\#}, \nu_{\$})$*, where*

- U *is a non-empty set of objects,*
- $I_{\#} : U \to P(U)$ *is an uncertainty function, where* $P(U)$ *denotes the power set of* U*,*
- $\nu_{\$} : P(U) \times P(U) \to [0,1]$ *is a rough inclusion function,*

and $\#, \$$ *denote vectors of parameters (the indexes* $\#, \$$ *will be omitted if it does not lead to misunderstanding.*

2.1 Uncertainty function

The uncertainty function defines for every object x, a set of objects described similarly to x. The set $I(x)$ is called the neighborhood of x (see, e.g., [18, 32]).

We assume that the values of the uncertainty function are defined using a *sensory environment*, i.e., a pair $(L, \| \cdot \|_U)$, where L is a set of formulas, called the *sensory formulas*, and $\| \cdot \|_U : L \longrightarrow P(U)$ is the *sensory semantics*. We assume that for any sensory formula α and any object $x \in U$ the information if $x \in \|\alpha\|_U$ holds is available. The set $\{\alpha : x \in \|\alpha\|_U\}$ is called the *signature of* x in AS and is denoted by $Inf_{AS}(x)$. For any $x \in U$ the *set* $\mathcal{N}_{AS}(x)$ of *neighborhoods of* x in AS is defined by $\{\|\alpha\|_U : x \in \|\alpha\|_U\}$ and from this set the neighborhood $I(x)$ is constructed. For example, $I(x)$ is defined by selecting an element from the set $\{\|\alpha\|_U : x \in \|\alpha\|_U\}$ or by $I(x) = \bigcap \mathcal{N}_{AS}(x)$. Observe that any sensory environment $(L, \| \cdot \|_U)$ can be treated as a parameter of I from the vector $\#$ (see Definition 1).

Let us consider two examples. Any decision table $DT = (U, A, d)$ [18] defines an approximation space $AS_{DT} = (U, I_A, \nu_{SRI})$, where, as we will see, $I_A(x) = \{y \in U : a(y) = a(x) \text{ for all } a \in A\}$. Any sensory formula is a descriptor, i.e., a formula of the form $a = v$ where $a \in A$ and $v \in V_a$ with the standard semantics $\|a = v\|_U = \{x \in U : a(x) = v\}$. Then, for any $x \in U$ its signature $Inf_{AS_{DT}}(x)$ is equal to $\{a = a(x) : a \in A\}$ and the neighborhood $I_A(x)$ is equal to $\bigcap \mathcal{N}_{AS_{DT}}(x)$. Another example can be obtained assuming that for any $a \in A$ there is given a tolerance relation $\tau_a \subseteq V_a \times V_a$ (see, e.g., [32]). Let $\tau = \{\tau_a\}_{a \in A}$. Then, one can consider a tolerance decision table $DT_\tau = (U, A, d, \tau)$ with tolerance descriptors $a =_{\tau_a} v$ and their semantics $\|a =_{\tau_a} v\|_U = \{x \in U : v\tau_a a(x)\}$. Any such tolerance decision table $DT_\tau = (U, A, d, \tau)$ defines the approximation space $AS_{DT_\tau} = (U, I_A, \nu_{SRI})$ with the signature $Inf_{AS_{DT_\tau}}(x) = \{a =_{\tau_a} a(x) : a \in A\}$ and the neighborhood $I_A(x) = \bigcap \mathcal{N}_{AS_{DT_\tau}}(x)$ for any $x \in U$.

The fusion of $\mathcal{N}_{AS_{DT_\tau}}(x)$ for computing the neighborhood of x can have many different forms; the intersection is only an example. One can also consider some more general uncertainty functions, e.g., with values in $P^2(U)$ [39]. For example, to compute the value of $I(x)$ first some subfamilies of $\mathcal{N}_{AS}(x)$ can be selected and next the family consisting of intersection of each such a subfamily is taken as the value of $I(x)$.

Note, that any sensory environment $(L, \| \cdot \|_U)$ defines an information system with the universe U of objects. Any row of such an information system for an object x consists of information if $x \in \|\alpha\|_U$ holds, for any sensory formula α. Let us also observe that in our examples we have used a simple sensory language defined by descriptors of the form $a = v$. One can consider a more general approach by taking, instead of the simple structure $(V_a, =)$, some other relational structures R_a with the carrier V_a for $a \in A$ and a signature τ. Then any formula (with one free variable) from a sensory language with the signature τ that is interpreted in R_a defines a subset $V \subseteq V_a$ and induces on the universe of objects a neighborhood consisting of all objects having values of the attribute a in the set V. Note, that this is the basic step in hierarchical modelling [38].

2.2 Rough inclusion function

One can consider general constraints which the rough inclusion functions should satisfy. Searching for such constraints initiated investigations resulting in creation and development of rough mereology (see, e.g., [27, 26] and the bibliography in [26]). In this subsection, we present only some examples of rough inclusion functions.

The rough inclusion function $\nu_\$: P(U) \times P(U) \to [0, 1]$ defines the degree of inclusion of X in Y, where $X, Y \subseteq U$.

In the simplest case it can be defined by (see, e.g., [32, 18]):

$$\nu_{SRI}(X, Y) = \begin{cases} \frac{card(X \cap Y)}{card(X)} & \text{if } X \neq \emptyset \\ 1 & \text{if } X = \emptyset. \end{cases}$$

This measure is widely used by the data mining and rough set communities. It is worth mentioning that Jan Łukasiewicz [12] was the first one who used this idea to estimate the probability of implications. However, rough inclusion can have a much more general form than inclusion of sets to a degree (see, e.g., [27, 26, 39]).

Another example of rough inclusion function ν_t can be defined using the standard rough inclusion and a threshold $t \in (0, 0.5)$ using the following formula:

$$\nu_t(X, Y) = \begin{cases} 1 & \text{if } \quad \nu_{SRI}(X, Y) \geq 1 - t \\ \frac{\nu_{SRI}(X,Y) - t}{1 - 2t} & \text{if } \ t \leq \nu_{SRI}(X, Y) < 1 - t \\ 0 & \text{if } \quad \nu_{SRI}(X, Y) \leq t. \end{cases}$$

The rough inclusion function ν_t is used in the variable precision rough set approach [49].

Another example of rough inclusion is used for function approximation [39] and relation approximation [41].

Then the inclusion function ν^* for subsets $X, Y \subseteq U \times U$, where $X, Y \subseteq \mathcal{R}$ and $\mathcal{R}$ is the set of reals, is defined by

$$\nu^* (X, Y) = \begin{cases} \frac{card(\pi_1(X \cap Y))}{card(\pi_1(X))} & \text{if } \pi_1(X) \neq \emptyset \\ 1 & \text{if } \pi_1(X) = \emptyset. \end{cases} \tag{1}$$

where π_1 is the projection operation on the first coordinate. Assume now, that X is a cube and Y is the graph $G(f)$ of the function $f : \mathcal{R} \longrightarrow \mathcal{R}$. Then, e.g., X is in the lower approximation of f if the projection on the first coordinate of the intersection $X \cap G(f)$ is equal to the projection of X on the first coordinate. This means that the part of the graph $G(f)$ is "well" included in the box X, i.e., for all arguments that belong to the box projection on the first coordinate the value of f is included in the box X projection on the second coordinate.

Usually, there are several parameters that are tuned in searching for a relevant rough inclusion function. Such parameters are listed in the vector $\#$. An example of such parameters is the threshold mentioned for the rough inclusion function used in the variable precision rough set model. We would like to mention some other important parameters. Among them are pairs $(L^*, \|\cdot\|_U^*)$ where L^* is an extension of L and $\|\cdot\|_U^*$ is an extension of $\|\cdot\|_U$, where $(L, \|\cdot\|_U)$ is a sensory environment. For example, if L consists of sensory formulas $a = v$ for $a \in A$ and $v \in V_a$ then one can take as L^* the set of descriptor conjunctions. For rule based classifiers we search in such a set of formulas for relevant patterns for decision classes. We present more detail in the following section.

2.3 Lower and upper approximations

The lower and the upper approximations of subsets of U are defined as follows.

Definition 2. *For any approximation space $AS_{\#,\$} = (U, I_\#, \nu_\$)$ and any subset $X \subseteq U$, the lower and upper approximations are defined by*

$LOW\left(AS_{\#,\$}, X\right) = \{x \in U : \nu_\$\left(I_\#\left(x\right), X\right) = 1\}$,

$UPP\left(AS_{\#,\$}, X\right) = \{x \in U : \nu_\$\left(I_\#\left(x\right), X\right) > 0\}$, *respectively.*

The lower approximation of a set X with respect to the approximation space $AS_{\#,\$}$ is the set of all objects, which can be classified with certainty as objects of X with respect to $AS_{\#,\$}$. The upper approximation of a set X with respect to the approximation space $AS_{\#,\$}$ is the set of all objects which can be possibly classified as objects of X with respect to $AS_{\#,\$}$.

Several known approaches to concept approximations can be covered using the discussed here approximation spaces, e.g., the approach given in [18], approximations based on the variable precision rough set model [49] or tolerance (similarity) rough set approximations (see, e.g., [32] and references in [32]).

Classification methods for concept approximation developed in machine learning and pattern recognition make it possible to decide for a given object if it belongs to the approximated concept or not [7]. The classification methods yield the decisions using only partial information about approximated concepts. This fact is reflected in the rough set approach by assumption that concept approximations should be defined using only partial information about approximation spaces. To decide if a given object belongs to the (lower or upper) approximation of a given concept the rough inclusion function values are needed. In the next section we show how such values necessary for classification making are estimated on the basis of available partial information about approximation spaces.

3 Concept Approximation by Partial Information about Approximation Spaces

In machine learning and pattern recognition [7] we often search for approximation of a concept $C \subset U^*$ in approximation space $AS^* = (U^*, I^*, \nu^*)$ having only a partial information about AS^* and C, i.e., information restricted to a sample $U \subset U^*$. Let us denote the restriction of AS^* to U by $AS = (U, I, \nu)$, i.e., $I(x) = I^*(x) \cap U$, $\nu(X,Y) = \nu^*(X,Y)$ for $x \in U, and X, Y \subseteq U$.

To decide if a given object x belongs to the lower approximation or the upper approximation of $C \subset U^*$, it is necessary to know the value $\nu^*(I^*(x), C)$. However, in case there is only partial information about the approximation space AS^* available, then one must make an estimation of such a value rather than its exact value. In machine learning, pattern recognition or data mining different heuristics are used for estimation of the values of ν^*. Using different heuristic strategies, values of another function ν' are computed and they are used for estimation of values of ν^*. Then, the function ν' is used for deciding if objects belong to C or not. Hence, we define an approximation of C in the approximation space $AS' = (U^*, I^*, \nu')$ rather than in AS^*. Usually, it is required that the approximations of $C \cap U$ in AS and AS' are close (or the same). If a new portion of objects extending the sample U to U_1 is received, then the closeness of approximations of C in the new approximation space $AS_1 = (U_1, I_1, \nu_1)$ (where I_1, ν_1 are obtained by restriction of I^*, ν^* to U_1) with approximations over AS' restricted to U_1 is verified. If the approximations are not close enough, then the definition of ν' is modified using new information about the extended sample. In this way, we gradually improve the quality of approximation of C on larger parts of the universe U^*. This idea is presented in Figure 1.

Now, we would like to explain in more detail a method for estimation of values $\nu^*(I^*(x), C)$. Let us consider an illustrative example. In the example we follow a method often used in rule based classifiers [39]. The method is based on the following steps. First, a set of patterns that are used as left hand sides of decision rules, is induced. Each pattern describes a set of objects in U^* with a satisfactory degree of inclusion to one of decision classes (C or $U^* - C$ for the binary decision). Next, for any object the set of all such patterns that are matched to a satisfactory degree by the given object is extracted. Finally, it is applied a conflict resolution strategy (e.g., voting) for resolving conflicts between votes for different decisions by the matched patterns.

We now present an illustrative example to describe this process more formally in the framework of approximation spaces. First, we assume that among

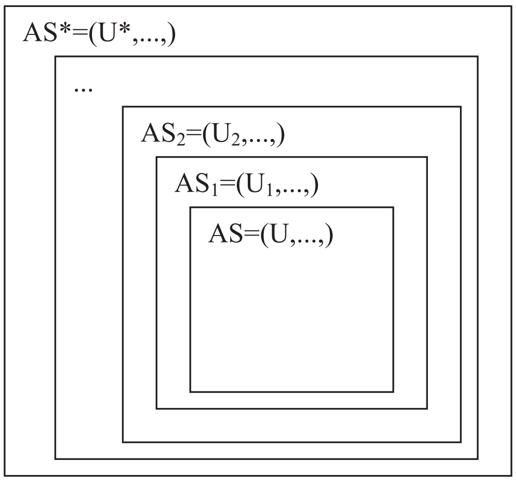

Fig. 1. Partial information about approximation space AS^*.

parameters of rough inclusion functions are pairs $(PAT, \|\cdot\|_{U^*})$, where PAT is a set of descriptor conjunctions over a set of condition attributes and $\|\cdot\|_{U^*} : PAT \longrightarrow P(U^*)$ is the semantics of patterns in U^*. Using such parameters we estimate the value $\nu^*(\|pat\|_{U^*}, C)$ by $\nu(\|pat\|_U,\ C \cap U)$ for any $pat \in PAT$ and we obtain, for a given threshold $deg \in [0,1]$, the set S_1 of all patterns pat such that $\nu(\|pat\|_U, C \cap U) \geq deg$, i.e., consisting of patterns "for" the concept C. In an analogous way we obtain the set S_2 of all patterns pat satisfying $\nu^*(\|pat\|_{U^*}, U^* - C) \geq deg$. S_2 consists of patterns "for" the complement $U^* - C$ of the concept C. Next, we estimate $\nu^*(I^*(x), \|pat\|_{U^*})$ for $pat \in S_i$, for $i = 1,2$. To do this we use our assumption on computing $I^*(x)$ for $x \in U^*$. We assume that the sensory formulas from L are descriptors $a = v$ over the condition attributes from a given set of condition attributes A with the semantics in U^* defined by $\|a = v\|_{U^*} = \{x \in U^* : a(x) = v\}$ for $a \in A$ and $v \in V_a$, where V_a is the value set of a. We also have $I^*(x) = \{y \in U^* : Inf_A(x) = Inf_A(y)\}$, where $Inf_A(x) = \{(a, a(x)) : a \in A\}$. Often, we estimate $\nu^*(I^*(x), \|pat\|_{U^*})$ using a matching strategy based on similarity of the syntactic description of x by $Inf_A(x)$ and the pattern pat. In this way we obtain for a given x the set S_i' of all patterns $pat \in S_i$ (for $i = 1,2$) such that $\nu(I^*(x) \cap U, \|pat\|_U) \geq deg_1$ where $deg_1 \in [0,1]$ is a given threshold. Finally, the estimation $\nu'(I^*(x), C)$ of the value $\nu^*(I^*(x), C)$ is obtained by application to the sets S_1', S_2' a conflict resolution strategy for resolving conflicts between patterns "for" and "against" the membership of x to C.

Usually, the function ν' is parameterized, e.g., by a threshold to which at least the patterns should be included into the decision classes. Also the discussed sets of patterns are among parameters of ν' tuned in the process of rule based classifier construction. Moreover, matching strategies used for estimation of matching degrees are usually parameterized and such parameters are also among tuned parameters of ν'. In machine learning, pattern recognition and data mining many different searching techniques have been developed for inducing concept approximations of the high quality. Among such components are relevant features, patterns, measures of closeness, model quality measures.

The approximation spaces defined above have been generalized in [39] to approximation spaces consisting of information granules.

4 Operations on approximation spaces

In this section, we introduce operations on approximation spaces called constrained sums of approximation spaces. On the basis of such operations we have developed a methodology for discovery of relevant patterns for complex concept approximations (see [3, 36, 37]), e.g., in hierarchical learning, ontology approximation, and spatio-temporal reasoning (see, e.g., [5, 6, 16, 38]). This methodology is also relevant for approximate reasoning in distributed environments.

We assume that for approximation spaces

$$AS_{\#,\$} = (U, I_{\#}, \nu_{\$})$$

considered in this section the following conditions are satisfied (see Section 2.1 and Section 2.2):

1. The values of the uncertainty function $I_{\#}$ are defined using the *sensory environment* $(L, \|\cdot\|_U)$ of $AS_{\#,\$}$, where L is a set of formulas, called the *sensory formulas*, and $\|\cdot\|_U : L \longrightarrow P(U)$ is the *sensory semantics*. The sensory environment is one of the components of the vector $\#$.
2. The values $I_{\#}(x)$ are defined by $\bigcap \mathcal{N}_{AS}(x)$ for any $x \in U$.
3. $\nu = \nu_{SRI}$ (i.e., ν is the standard rough inclusion, see Section 2)
4. Only a partial information about the approximation space $AS_{\#,\$}$ is given, i.e., the restriction of $AS_{\#,\$}$ to a subset $U_o \subseteq U$.
5. The values of the rough inclusion function $\nu_{\$}$ are estimated using a pair $(L^*, \|\cdot\|_U^*)$ where L^* is an extension of L and $\|\cdot\|_U^*$ is na extension of $\|\cdot\|_U$, where $(L, \|\cdot\|_U)$ is the sensory environment of $AS_{\#,\$}$. The pair $(L^*, \|\cdot\|_U^*)$ is one of the components of the vector $\$$. The formulas from L^* are patterns that are used for estimation of values of $\nu_{\$}$ (see Section 3).

Now, the operations on approximation spaces, by analogy to the constrained sums of information systems [3, 36, 37], can be defined as follows. To simplify notation we consider only the case of binary operations.

For approximation spaces AS^i for $i = 1, 2$ we consider the class

$$CONSTRAINT(AS^1, AS^2)$$

of all approximation spaces $AS = (U, I, \nu)$ satisfying the following conditions:

1. For estimation of values of ν an extension $(L^*, \|\cdot\|_U^*)$ of $(L^{*,i}, \|\cdot\|_U^{*,i})$, where $i = 1, 2$ is used. An extension is satisfying the conditions: $L^{*,1} \cup L^{*,2} \subseteq L$ and $\|\alpha\|_U = \|\alpha\|_U^i$ for $\alpha \in L^i$, where $i = 1, 2$. The formulas from $L^* - (L^{*,1} \cup L^{*,2})$ are called constraints. We also assume that any constraint α is a boolean combination of formulas from $L^{*,1} \cup L^{*,2}$, e.g., disjunction of formulas $\alpha_1 \wedge \alpha_2$ for $\alpha_1 \in L^{*,1}$ and $\alpha_2 \in L^{*,2}$.
2. The sensory environment of AS is defined by $(L, \|\cdot\|_U)$.

Any operation o on approximation spaces such that

$$o(AS^1, AS^2) \in CONSTRAINT(AS^1, AS^2)$$

for any approximation spaces AS^1, AS^2 from a generic set $\mathcal{AS}$ of approximation spaces, is called the constrained sum.

4.1 Hierarchical Learning

In hierarchical learning we consider the space $SPACE(\mathcal{AS}, \mathcal{F})$ generated from $\mathcal{AS}$ by the set $\mathcal{F}$ of constrained sums. Searching for relevant approximation spaces from $SPACE(\mathcal{AS}, \mathcal{F})$ is making it possible to discover relevant patterns for concept approximation [5, 6, 16]. Constrained sums of approximation spaces are tools for modelling patterns that are more relevant for approximation of concepts than patterns defined by arguments of the constrained sum.

Assume that approximation spaces AS^1, AS^2 are used for approximation of concepts C_1, C_2 and that the dependency *if* C_1 and C_2 *then* C holds. Note, that usually many such dependencies should be considered for the concept approximation. However, for simplicity of presentation we consider only one.

If the concept C is, in a sense, not *to far* from C_1 and C_2 than one can search in $SPACE(\mathcal{AS}, \mathcal{F})$ for a constrained sum $o(AS^1, AS^2)$ relevant for the concept C approximation. If $AS = o(AS^1, AS^2)$ than patterns defined in AS are more general than the conjunction of sensory formulas defined by AS^1, AS^2. This happens because new patterns are defined by joining patterns of AS^1, AS^2 relative to constraints. Let us recall that patterns of AS^1, AS^2 belong to the extension of the set of sensory formulas of AS^1, AS^2 (see the definition of $CONSTRAINT(AS^1, AS^2)$). Hence, sensory formulas of AS are conjunction of patterns of AS^1, AS^2 and constraints. By allowing to use patterns instead of sensory formulas in joining AS^1, AS^2 we define a searching space for relevant patterns and from such a space are extracted relevant joins (relative to constraints) of patterns for approximation of concepts.

If the concept C is *far* from C_1 and C_2 then we use a hierarchy of dependencies between concepts from domain knowledge in searching for the relevant approximation space for C. The approximation of C is obtained using hierarchical learning [5, 6, 16]. We assume that (1) the generic concepts C_1 and C_2 on the lowest level of the hierarchy can be approximated by some generic approximation spaces, (2) the concepts on a given level that are not generic follow form concepts on the lower level, and (3) the relevant approximation spaces for concepts on a given level of the hierarchy can be discovered using relevant constrained sums of approximation spaces from the previous hierarchy level. Then we proceed as follows. Starting from some generic concepts and approximation spaces relevant for them we construct approximation spaces for concepts on the first level in hierarchy that follow from these generic concepts and can be approximated by relevant constrained sums over these approximation spaces. Next, we perform the same procedure for the recently approximated concepts and the concepts on the next level of hierarchy. We continue the procedure until the target concept is approximated.

Let us consider one more example of applications of constrained sums of approximation spaces for approximation of dependency between vague concepts. This problem is important in ontology approximation [34, 40]. Any concept from the left hand side of a given vague dependency is called its premise and the dependency conclusion is the concept from the right hand side of the dependency. The approximation of a given vague dependency is defined by a method which allows for any object to compute the arguments "for" and "against" its membership to the dependency conclusion on the basis of analogous arguments relative to the dependency premisses [40]. Any argument "for" or "against" is a compound information granule (pattern) consisting of a pattern together with a degree to which (at least) this pattern is included to the concept and a degree to which (at least) the analyzed object is included to the pattern. Such arguments "for" and "against" that are relevant for the dependency conclusion are constructed from the arguments "for" and "against" for the dependency premisses by using constrained sums. Such constructions are called the local schemes. Any local scheme (production rule) (see, e.g., [35]) or rough mereological connective (see, e.g., [28]) yields the fusion result of arguments for premisses that is next taken as the argument for the dependency conclusion. By composition of local schemes more advanced fusion schemes are obtained, called approximate reasoning schemes (AR schemes) (see, e.g., [5, 35, 28, 38]). They show how the arguments from premisses of dependencies are fused to arguments for more compound concepts derived in a given ontology from premisses. AR schemes can correspond to different parts of complex spatio-temporal objects. Hence, there is a need for composing AR schemes for parts into AR schemes for objects composed from these parts [38].

5 Reinforcement Learning

By way of another illustration of the utility of approximation spaces, a rough set approach to reinforcement learning is briefly considered in this section. The study of reinforcement learning carried out in the context of approximation spaces is outgrowth of recent work on approximate reasoning and intelligent systems (see, e.g., [24, 25, 23, 30, 31, 20, 22]). An overview of a Monte Carlo approach to reinforcement learning with approximation spaces is given in [21]. The basic problem that provides a setting for reinforcement learning is formulated by [29]: a system is required to interact with its environment to achieve a particular task or goal, and based on the feedback about the current state of the environment, what action should the system perform next? Reinforcement learning itself is the act of learning the correct action to take in a specific situation based on feedback obtained from the environment [44].

Feedback is in the form of a numerical reward that results from an action performed by an agent. Specifically, reinforcement learning can be divided into *off-line* and *on-line* learning. Off-line learning is similar to the idea of a student learning by instruction from a teacher. In effect, the agent is taught what it needs to know before venturing into the environment in which it is to operate. In contrast, on-line learning resembles an infant learning to walk. Learning occurs in real-time in which the agent is exploring its environment and constantly adding to its experience in order to make better decisions in the future. Learning techniques are typically applied to stationary or non-stationary models of the environment. In stationary models all the state transition probabilities are fixed, whereas, in non-stationary models they change over time.

Let $AS_{DT,B} = (U_{beh}, I_B, \nu_B)$ denote an approximation space defined in the context of a decision system $DT = (U_{beh}, A, d)$, where U_{beh} is a non-empty set of behaviors, A is a set of swarmbot behavior features, $B \subseteq A$, and d is a distinguished attribute representing a swarmbot decision. Let $D = \{x \in U : d(x) = 1\}$, where $d(x) = 1$ specifies that behavior x has been accepted by a swarmbot. Assume that $I_B : U_{beh} \rightarrow P(U_{beh})$ is used to compute $B_*D = LOW\,(AS_{DT,B}, D)$. We also assume that among condition attributes there is an attribute $Action$ such that V_{Action} is the set of possible actions in the considered system and $Action(x)$ denotes the action performed in state x by the system. Moreover, we assume that for a condition attribute $Time \in A$ it is recorded information about the time in which the state has been observed. We use the notation s_t to denote a pair $(s, Time(s))$. Further, let B_*D represent a standard for swarmbot behaviors, and let $B_{ac}(x)$ be a block in the partition of U_{beh} containing x relative to action ac (i.e., $B_{ac}(x)$ contains behaviors for a particular action ac that are equivalent to x). The block $B_{ac}(x)$ where $ac \in V_{Action}$ is defined in (2).

$$B_{ac}(x) = \begin{cases} \{y \in U_{beh} : xIND\,(B \cup \{Action\})\, y\} & \text{if } Action(x) = ac \\ \emptyset & \text{otherwise.} \end{cases} \quad (2)$$

Then we can measure the inclusion of $B_{ac}(x)$ into B_*D as in (3).

$$\nu(B_{ac}\,(x)\,, B_*D) = \begin{cases} \nu_{SRI}(B_{ac}\,(x)\,, B_*D) & \text{if } B_{ac}(x) \neq \emptyset \\ 0 & \text{otherwise,} \end{cases} \quad (3)$$

where ν_{SRI} is the standard rough inclusion function (see Section 2.2).

B_*D represents certain knowledge about the behaviors in D. For this reason, B_*D provides a useful behavior standard or behavior norm in gaining knowledge about the proximity of behaviors to what is considered normal. The term *normal* applied to a set of behaviors denotes forms of behavior that have been accepted. The introduction of some form of behavior standard makes it possible to measure the inclusion of blocks of equivalent action-specific behaviors in the set of those behaviors that are part of a standard. The framework provided by an approximation space makes it possible to derive pattern-based rewards, which are used by swarms that learn to choose actions in response to perceived states of their environment (see, e.g., Figure 2). The notation $\bar{r}_{ac,t}$ denotes an average rough inclusion value computed within the context of an approximation space using (3) as shown in (4).

$$\bar{r}_{ac,t} = \sum_{i=0}^{n} \nu(B_{ac}(x_{t-i}), B_* D)/n \tag{4}$$

where $T = (t - n, \ldots, t)$ denotes the time window of the length n, $x_{t-n}, \ldots, x_t$ are states observed in this time window (i.e., in an episode of the system).

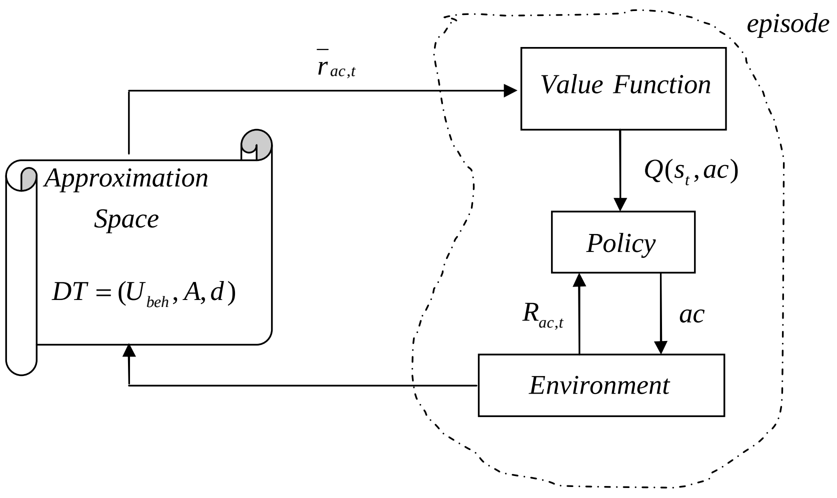

Fig. 2. Reinforcement learning framework

The notation *pc* in Figure 2 denotes proximate cause, which is one of the four *whys* introduced by Niko Tinbergen [45] to explain observed behavior (see, e.g., [19, 21], where *pc* is explained in more detail in the context of approximation spaces). The decision table $DT = (U_{beh}, A, d)$ in Figure 2 provides a record of behavior patterns observed during different episodes in the life of system.

The notation $Q(s_t, ac)$ in Figure 2 denotes the value of an action ac in state s at time t, i.e., in s_t.

There are different strategies for computing $Q(s_t, ac)$. They make it possible to estimate how successful the performance of an action ac in s_t can be.

For example, let $W_{ac,t}$ denote an average of the weights over a time window (see (5)).

$$W_{ac,t} = \frac{\sum_{i=1}^{n} \bar{r}_{ac,t-i}}{n}, \tag{5}$$

Let us now assume that $R_{ac,t}$ denotes the sum of the rewards (returns) during an episode for the action ac over the time window ending at t. Then the value $Q(s_t, ac)$ of ac in s_t is defined by

$$Q(s_t, ac) = \sum_{i=1}^{n} \frac{\bar{r}_{ac,t-i}}{W_{ac,t-i}} \left[R_{ac,t-i} - Q(s_{t-i}, ac) \right]. \tag{6}$$

An analogous strategy has been successfully used in a new form of Monte Carlo off-policy reinforcement learning (see, e.g., [21]).

Conclusions

We discussed the approximation of concepts using the rough set approach. In particular, the role of approximation spaces and operations on approximation spaces, called constrained sums, in hierarchical learning has been emphasized.

In our project we are developing evolutionary strategies searching for relevant approximation spaces for concept approximation of a given ontology [34] of concepts. We also investigate properties of evolutionary strategies for constructing sequences of approximation spaces in adaptive approximation of concepts.

We also plan to use the methodology for concept approximation modelling by constrained sums in multiagent systems [11]. Constructing of relevant constrained sums requires negotiations and conflict resolution between agents constructing approximation spaces for different, e.g., local and global goals. Hence, e.g., strategies for formation of coalitions in cooperative searching for relevant approximation spaces for different local and global goals related to concept approximation are needed.

Acknowledgements

The research has been supported by the grants 3 T11C 002 26 and 4 T11C 014 25 from Ministry of Scientific Research and Information Technology of the Republic of Poland. This research has also been supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) grant 185986 and grant T247 from Manitoba Hydro.

References

1. Albus, J.S., Meystel, A. M.: *Engineering of mind: An introduction to the science of intelligent systems.* John Wiley, New York, 2001.
2. Anderson, J.R.: *Rules of the Mind.* Lawrence Erlbaum, Hillsdale, NJ, 1993.

3. Barwise, J., Seligman, J.: *Information Flow: The Logic of Distributed Systems.* Cambridge University Press Tracts in Theoretical Computer Science **44**, 1997.
4. Bazan, J., Peters, J., F., Skowron, A.: Behavioral pattern identification through rough set modelling. *Proceedings of the Tenth International Conference on Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing* (RSFDGrC 2005), August 31st - September 3rd, 2005, University of Regina, Canada (to appear).
5. Bazan, J., Skowron, A.: Classifiers based on approximate reasoning schemes. In: Dunin-Keplicz, B., Jankowski A., Skowron A., and Szczuka M., (Eds.), *Monitoring, Security, and Rescue Tasks in Multiagent Systems MSRAS*, Advances in Soft Computing. Springer, Heidelberg, 2005, 191–202.
6. Bazan, J., Nguyen, S. Hoa, Nguyen, H. Son, Skowron, A.: Rough set methods in approximation of hierarchical concepts. *Proc. of RSCTC'2004*, LNAI **3066**, Springer, Heidelberg, 2004, 346–355.
7. Hastie, T., Tibshirani, R., Friedman, J.: *The Elements of Statistical Learning.* Springer-Verlag, Heidelberg, 2003.
8. Kieras, D., Meyer, D.E.: An overview of the EPIC architecture for cognition and performance with application to human-computer interaction. *Human-Computer Interaction* **12**: 391-438, 1997.
9. Langley, P., Laird, J.E.: Cognitive architectures: Research issues and challenges. *Technical Report*, Institute for the Study of Learning and Expertise, Palo Alto, CA, 2002.
10. Laird, J.E., Newell, A.: Rosenbloom P.S.: Soar: An architecture for general intelligence. *Artificial Intelligence* **33**: 1-64, 1987.
11. Luck, M., McBurney, P., Preist, Ch.: *Agent Technology: Enabling Next Generation. A Roadmap for Agent Based Computing*, Agent Link, 2003.
12. Łukasiewicz, J.: Die logischen Grundlagen der Wahrscheinilchkeitsrechnung, Kraków 1913. In: Borkowski, L. (ed.), *Jan Łukasiewicz - Selected Works.* North Holland, Amstardam, Polish Scientific Publishers, Warsaw, 1970.
13. Meyer, D. E., Kieras, D. E.: A computational theory of executive cognitive processes and multiple-task performance: Part 1. Basic mechanisms. *Psychological Review* **104**: 3–65, 1997.
14. Meyer, D. E., Kieras, D. E.: A computational theory of executive control processes and human multiple-task performance: Part 2. Accounts of Psychological Refractory-Period Phenomena. *Psychological Review* **104**: 749–791, 1997.
15. Newell, A.: (1990). *Unified Theories of Cognition.* Cambridge, Harvard University Press, MA, 1980.
16. Nguyen, S. Hoa, Bazan, J., Skowron, A., Nguyen, H. Son: Layered learning for concept synthesis. *Transactions on Rough Sets I: LNCS Journal Subline*, LNCS **3100**, Springer, Heidelberg, 2004, 187–208
17. Pawlak, Z.: Rough sets, *International J. Comp. Inform. Science* **11**: 341–356, 1982.
18. Pawlak, Z.: *Rough Sets. Theoretical Aspects of Reasoning about Data.* Kluwer Academic Publishers, Dordrecht, 1991.
19. Peters, J.F.: Rough ethology: Towards a biologically-inspired study of collective behavior in intelligent systems with approximation spaces. LNAI **3400**, *Transactions on Rough Sets* **III**: 153–174, Springer, Heidelberg, 2005.
20. Peters, J. F. : Approximation space for intelligent system design patterns. *Engineering Applications of Artificial Intelligence* **17**(4): 1–8, 2004.
21. Peters, J.F., Henry, C.: Reinforcement learning with approximation spaces. *Fundamenta Informaticae*, 2005, submitted.
22. Peters, J. F., Skowron, A., Stepaniuk, J., Ramanna, S.: Towards an ontology of approximate reason. *Fundamenta Informaticae* **51**(1-2), 157–173, 2002.

23. Pal, S.K., Polkowski, L., Skowron, A. (Eds.): *Rough-Neural Computing: Techniques for Computing with Words*. Springer-Verlag, Berlin, 2004.
24. Polkowski, L., Skowron, A.: Rough mereology: A new paradigm for approximate reasoning. *International Journal Approximate Reasoning* **15**(4): 333–365, 1996.
25. Polkowski, L., Skowron, A.: Approximate reasoning about complex objects in distributed systems: Rough mereological foundations. In: W. Pedrycz, J.F. Peters (Eds.), Computational Intelligence in Software Engineering. *Advances in Fuzzy Systems-Applications and Theory* **16**, World Scientific, Singapore, 1998, 237–267.
26. Polkowski, L.: *Rough Sets: Mathematical Foundations*. Advances in Soft Computing, Physica-Verlag, Heidelberg, 2002.
27. Polkowski, L., Skowron, A.: Rough mereology: A new paradigm for approximate reasoning. *Journal of Approximate Reasoning* **15**(4): 333–365, 1996.
28. Polkowski, L., Skowron, A.: Towards adaptive calculus of granules. In: [47], 201–227.
29. Rummery, G.A.: *Problem Solving with Reinforcement Learning*. Ph.D. Thesis, Cambridge University, 1995.
30. Skowron,A.: Toward intelligent systems: Calculi of information granules. *Bulletin of the Rough Set Society*, **5**(1-2): 9–30, 2001.
31. Skowron, A., Swiniarski, R.W.: Information granulation and pattern recognition. In: [23], 2004, 599–636.
32. Skowron, A., Stepaniuk, J.: Tolerance approximation spaces. *Fundamenta Informaticae* **27**: 245–253, 1996.
33. Skowron, A., Stepaniuk, J.: Information granules: Towards foundations of granular computing. *International Journal of Intelligent Systems* **16**(1): 57–86, 2001.
34. Staab, S., Studer, R., (Eds.): *Handbook on Ontologies*. International Handbooks on Information Systems, Springer, Heidelberg, 2004.
35. Skowron, A., Stepaniuk, J.: Information granules and rough-neural computing. In: [23], 43–84.
36. Skowron, A., Stepaniuk, J., Peters, J.F.: Rough sets and infomorphisms: Towards approximation of relations in distributed environments. *Fundamenta Informaticae* **54**(1-2): 263–277, 2003.
37. Skowron, A., Stepaniuk, J.: Constrained sums of information systems. In: *Proc. RSCTC 2004*, LNCS **3066**, Springer, Heidelberg, 2004, 300–309.
38. Skowron, A., Synak, P., Complex patterns. *Fundamenta Informaticae* **60**(1-4): 351–366, 2004.
39. Skowron, A., Swiniarski, R., Synak, P: Approximation spaces and information granulation. *Transactions on Rough Sets III: LNCS Journal Subline*, LNCS **3400**, Springer, Heidelberg, 2005, 175–189.
40. Skowron, A., Stepaniuk, J. (2005). Ontological Framework for Approximation. *Proceedings of the Tenth International Conference on Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing* (RSFDGrC 2005), August 31st - September 3rd, 2005, University of Regina, Canada (to appear).
41. Stepaniuk, J.: Rough relations and logics. In: L. Polkowski, A. Skowron (Eds.), Rough Sets in Knowledge Discovery 1. Methodology and Applications, Physica Verlag, Heidelberg, 1998, 248–260.
42. Stepaniuk, J.: Knowledge discovery by application of rough set models. In: L. Polkowski, S. Tsumoto, T.Y. Lin (Eds.), *Rough Set Methods and Applications. New Developments in Knowledge Discovery in Information Systems*, Physica–Verlag, Heidelberg, 2000, 137–233.
43. Stone, P.: *Layered Learning in Multi-Agent Systems: A Winning Approach to Robotic Soccer*. MIT Press, Cambridge, MA, 2000.

44. Sutton, R.S., Barto, A.G.: *Reinforcement Learning: An Introduction.* The MIT Press, Cambridge, MA, 1998.
45. Tinbergen, N.: On aims and methods of ethology, *Zeitschrift für Tierpsychologie* **20**: 410–433, 1963.
46. Veloso, M.M., Carbonell, J.G.: Derivational analogy in PRODIGY: Automating case acquisition, storage, and utilization. *Machine Learning* **10**: 249-278, 1993.
47. Zadeh, L.A., Kacprzyk, J. (Eds.): *Computing with Words in Information/Intelligent Systems* **1-2**, Physica-Verlag, Heidelberg, 1999.
48. Zadeh, L.A.: A new direction in AI: Toward a computational theory of perceptions. *AI Magazine* **22**(1): 73–84, 2001.
49. Ziarko, W., Variable precision rough set model, *Journal of Computer and System Sciences* **46**: 39–59, 1993.