A Comprehensive Implementation of Conceptual Spaces

Lucas Bechberger* and Kai-Uwe Kühnberger

Institute of Cognitive Science, Osnabrück University, Osnabrück, Germany
lucas.bechberger@uni-osnabrueck.de, kai-uwe.kuehnberger@uni-osnabrueck.de

Abstract. The highly influential framework of conceptual spaces provides a geometric way of representing knowledge. Instances are represented by points and concepts are represented by regions in a (potentially) high-dimensional space. Based on our recent formalization, we present a comprehensive implementation of the conceptual spaces framework that is not only capable of representing concepts with inter-domain correlations, but that also offers a variety of operations on these concepts.

1 Introduction

One common criticism of symbolic AI approaches is that the symbols they operate on do not contain any meaning: For the system, they are just arbitrary tokens that can be manipulated in some way. This lack of inherent meaning in abstract symbols is called the symbol grounding problem [18]. One approach towards solving this problem is to devise a grounding mechanism that connects abstract symbols to the real world, i.e., to perception and action.

The cognitive framework of conceptual spaces [16,17] attempts to bridge this gap between symbolic and subsymbolic AI by proposing an intermediate conceptual layer based on geometric representations. A conceptual space is spanned by a number of quality dimensions that are based on perception and/or subsymbolic processing. Regions in this space correspond to concepts and can be referred to as abstract symbols.

The framework of conceptual spaces has been highly influential in the last 15 years within cognitive science and cognitive linguistics [14,15,27]. It has also sparked considerable research in various subfields of artificial intelligence, ranging from robotics and computer vision [11,12] over the semantic web [2] to plausible reasoning [13,24].

Most practical implementations of the conceptual spaces framework are rather ad-hoc and tailored towards a specific application. They tend to ignore important aspects of the framework and should thus be regarded as only partial implementations of the framework. Furthermore, these implementations are usually not publicly accessible, which greatly reduces their value for the research community.

---

*Corresponding author, ORCID: 0000-0002-1962-1777

In this paper, we present a thorough and comprehensive implementation of the conceptual spaces framework which is based on our formalization reported in [7] and [8]. Its source code is publicly available on GitHub to researchers anywhere in the world. Instead of investing a considerable amount of time into developing their own implementation, researchers can use our implementation off-the-shelf and focus on their specific application scenario.

The remainder of this paper is structured as follows: Section 2 introduces the framework of conceptual spaces and our formalization. In Section 3, we give an overview of our implementation and in Section 4 we illustrate its usage. Section 5 summarizes related work and Section 6 concludes the paper.

2   Conceptual Spaces

This section presents the cognitive framework of conceptual spaces as described in [16] and summarizes our formalization as reported in [7] and [8].

2.1   Dimensions, Domains, and Distance

A conceptual space is spanned by a set D of so-called "quality dimensions". Each of these dimensions $d \in D$ represents a way in which two stimuli can be judged to be similar or different. Examples for quality dimensions include temperature, weight, time, pitch, and hue. The distance between two points x and y with respect to a dimension d is denoted as $|x_d - y_d|$.

A domain $\delta \subseteq D$ is a set of dimensions that inherently belong together. Different perceptual modalities (like color, shape, or taste) are represented by different domains. The color domain for instance consists of the three dimensions hue, saturation, and brightness. Distance within a domain δ is measured by the weighted Euclidean metric d_E.

The overall conceptual space CS is defined as the product space of all dimensions. Distance within the overall conceptual space is measured by the weighted Manhattan metric d_M of the intra-domain distances. This is supported by both psychological evidence [5,25] and mathematical considerations [3]. Let Δ be the set of all domains in CS. The combined distance d_C^{Δ} within CS is defined as follows:

$$d_C^{\Delta}(x, y, W) = \sum_{\delta \in \Delta} w_\delta \cdot \sqrt{\sum_{d \in \delta} w_d \cdot |x_d - y_d|^2}$$

The parameter $W = \langle W_\Delta, \{W_\delta\}_{\delta \in \Delta} \rangle$ contains two parts: W_Δ is the set of positive domain weights w_δ with $\sum_{\delta \in \Delta} w_\delta = |\Delta|$. Moreover, W contains for each domain $\delta \in \Delta$ a set W_δ of positive dimension weights w_d with $\sum_{d \in \delta} w_d = 1$.

The similarity of two points in a conceptual space is inversely related to their distance. This can be written as follows :

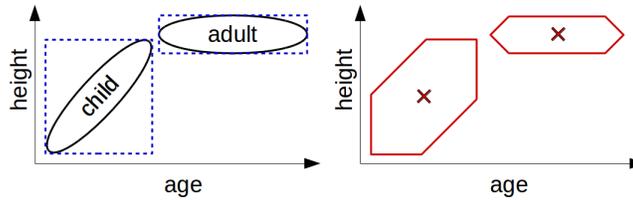$$Sim(x, y) = e^{-c \cdot d(x,y)} \quad \text{with a constant } c > 0 \text{ and a given metric } d$$

Fig. 1. Left: Intuitive way to define regions for the concepts of "adult" and "child" (solid) as well as representation by using convex sets (dashed). Right: Representation by using star-shaped sets with central points marked by crosses.

Betweenness is a logical predicate $B(x, y, z)$ that is true if and only if y is considered to be between x and z. It can be defined based on a given metric d:

$$B_d(x, y, z) : \iff d(x, y) + d(y, z) = d(x, z)$$

The betweenness relation based on d_E results in the line segment connecting the points x and z, whereas the betweenness relation based on d_M results in an axis-parallel cuboid between the points x and z. One can define convexity and star-shapedness based on the notion of betweenness:

Definition 1. *(Convexity)*
A set $C \subseteq CS$ is convex *under a metric d* $: \iff$
$\qquad \forall x \in C, z \in C, y \in CS : (B_d(x, y, z) \to y \in C)$

Definition 2. *(Star-shapedness)*
A set $S \subseteq CS$ is star-shaped *under a metric d with respect to a set $P \subseteq S$* $: \iff$
$\qquad \forall p \in P, z \in S, y \in CS : (B_d(p, y, z) \to y \in S)$

2.2 Properties and Concepts

Gärdenfors [16] distinguishes properties like "red", "round", and "sweet" from full-fleshed concepts like "apple" or "dog" by observing that properties can be defined on individual domains (e.g., color, shape, taste), whereas full-fleshed concepts involve multiple domains. Each domain involved in representing a concept has a certain importance, which is reflected by so-called "salience weights". Another important aspect of concepts are the correlations between the different domains, which are important for both learning [10] and reasoning [22, Ch 8].

Based on the principle of cognitive economy, Gärdenfors argues that both properties and concepts should be represented as convex sets. However, this convexity assumption has recently been criticized [19] and we demonstrated in [7] that one cannot geometrically encode correlations between domains when using convex sets: The left part of Figure 1 shows two domains, age and height, and the concepts of child and adult. The solid ellipses illustrate the intuitive way of defining these concepts. As domains are combined with the Manhattan metric, a convex set corresponds in this case to an axis-parallel cuboid. One can easily see

that this convex representation (dashed rectangles) is not satisfactory, because the correlation of the two domains is not encoded. We therefore proposed in [7] to relax the convexity criterion and to use star-shaped sets, which is illustrated in the right part of Figure 1. This enables a geometric representation of correlations while still being only a minimal departure from the original framework.

We have based our formalization on axis-parallel cuboids that can be described by a triple $\langle \Delta_C, p^-, p^+ \rangle$ consisting of a set of domains Δ_C on which this cuboid C is defined and two points p^- and p^+, such that

$$x \in C \iff \forall \delta \in \Delta_C : \forall d \in \delta : p_d^- \leq x_d \leq p_d^+$$

These cuboids are convex under d_C^Δ. It is also easy to see that any union of convex sets that have a non-empty intersection is star-shaped [26]. We define the core of a concept as follows:

Definition 3. *(Simple star-shaped set)*
A simple star-shaped set S is described as a tuple $\langle \Delta_S, \{C_1, \ldots, C_m\} \rangle$. $\Delta_S \subseteq \Delta$ is a set of domains on which the cuboids $\{C_1, \ldots, C_m\}$ (and thus also S) are defined. Moreover, we require that the central region $P := \bigcap_{i=1}^m C_i \neq \emptyset$. Then the simple star-shaped set S is defined as

$$S := \bigcup_{i=1}^m C_i$$

In order to represent imprecise concept boundaries, we use fuzzy sets [9,28,29]. A fuzzy set is characterized by its membership function $\mu : CS \rightarrow [0,1]$ which assigns a degree of membership to each point in the conceptual space. The membership of a point to a fuzzy concept is based on its maximal similarity to any of the points in the concept's core:

Definition 4. *(Fuzzy simple star-shaped set)*
A fuzzy simple star-shaped set $\widetilde{S}$ is described by a quadruple $\langle S, \mu_0, c, W \rangle$ where $S = \langle \Delta_S, \{C_1, \ldots, C_m\} \rangle$ is a non-empty simple star-shaped set. The parameter $\mu_0 \in (0,1]$ controls the highest possible membership to $\widetilde{S}$ and is usually set to 1. The sensitivity parameter $c > 0$ controls the rate of the exponential decay in the similarity function. Finally, $W = \langle W_{\Delta_S}, \{W_\delta\}_{\delta \in \Delta_S} \rangle$ contains positive weights for all domains in Δ_S and all dimensions within these domains, reflecting their respective importance. We require that $\sum_{\delta \in \Delta_S} w_\delta = |\Delta_S|$ and that $\forall \delta \in \Delta_S : \sum_{d \in \delta} w_d = 1$. The membership function of $\widetilde{S}$ is then defined as follows:

$$\mu_{\widetilde{S}}(x) = \mu_0 \cdot \max_{y \in S}(e^{-c \cdot d_C^\Delta(x,y,W)})$$

The sensitivity parameter c controls the overall degree of fuzziness of $\widetilde{S}$ by determining how fast the membership drops to zero. The weights W represent not only the relative importance of the respective domain or dimension for the
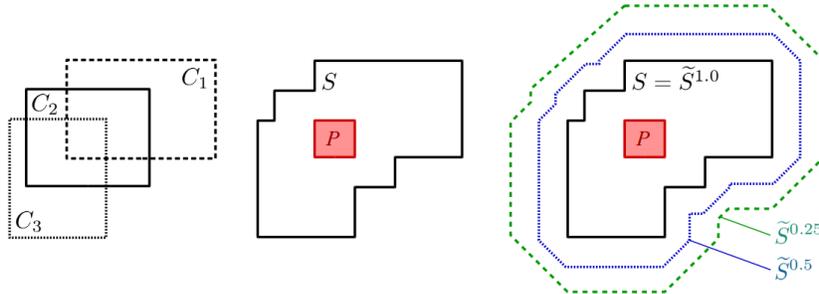
Fig. 2. Left: Three cuboids C_1, C_2, C_3 with nonempty intersection. Middle: Resulting simple star-shaped set S based on these cuboids. Right: Fuzzy simple star-shaped set $\widetilde{S}$ based on S with three α-cuts for $\alpha \in \{1.0, 0.5, 0.25\}$.

represented concept, but they also influence the relative fuzziness with respect to this domain or dimension. Note that if $|\Delta_S| = 1$, then $\widetilde{S}$ represents a property, and if $|\Delta_S| > 1$, then $\widetilde{S}$ represents a concept. Figure 2 illustrates these definitions (the x and y axes are assumed to belong to different domains and are combined with d_M using equal weights).

2.3 Operations on Concepts

Our formalization provides a number of operations, which can be used to create new concepts from existing ones and to describe relations between concepts. We summarize them here only briefly, more details can be found in [7] and [8].

Intersection. The intersection of two concepts can be interpreted as the logical "and" – e.g., intersecting the property "green" with the concept "banana" results in the set of all objects that are both green and bananas. The intersection of two concepts is defined as follows: The core of the resulting concept is the intersection of the highest intersecting α-cuts[1] of both original concepts, i.e., $S' = \widetilde{S}_1^{\alpha'} \cap \widetilde{S}_2^{\alpha'}$ with $\alpha' = \max\{\alpha \in [0,1] : \widetilde{S}_1^{\alpha} \cap \widetilde{S}_2^{\alpha} \neq \emptyset\}$. As this intersection is not guaranteed to be a valid core, we approximate it by cuboids. If these cuboids do not intersect, we compute the arithmetic mean of the cuboids' centers and extend each cuboid, such that it contains this central point. We use α' as the resulting concept's μ_0 parameter, set $c' := \min(c^{(1)}, c^{(2)})$, and derive a new set of weights by interpolating between the two original sets of weights.

Union. The union of two concepts can be used to construct more abstract categories (e.g., defining "fruit" as the union of "apple", "banana", "coconut", etc.). We define the union of two concepts as follows: The core of the new concept is

---

[1] The α-cut of $\widetilde{S}$ is defined as $\widetilde{S}^{\alpha} := \{x \in CS \mid \mu_{\widetilde{S}}(x) \geq \alpha\}$.

the union of the original cores. If it is not star-shaped, the same repair mechanism as for the intersection is used. We further set $\mu_0' := \max(\mu_0^{(1)}, \mu_0^{(2)})$ and compute c' and W' as described for the intersection.

Subspace Projection. Projecting a concept onto a subspace corresponds to focusing on certain domains while ignoring others. For instance, projecting the concept "apple" onto the color domain results in a property that describes the typical color of apples. The projection of a concept $\widetilde{S}$ onto domains $\Delta_{S'} \subseteq \Delta_S$ is defined by projecting its core S onto $\Delta_{S'}$, removing all weights that are irrelevant for $\Delta_{S'}$, and keeping μ_0 and c the same.

Axis-Parallel Cut. One can split a concept $\widetilde{S}$ into two parts (e.g., during a clustering process) by selecting a value v on a dimension d and by splitting each cuboid $C \in S$ into $C^{(+)} := \{x \in C \mid x_d \geq v\}$ and $C^{(-)} := \{x \in C \mid x_d \leq v\}$.[2] It is easy to see that $S^{(+)} := \bigcup C_i^{(+)}$ and $S^{(-)} := \bigcup C_i^{(-)}$ are still valid cores. The parameters μ_0, c, and W remain unchanged when defining $\widetilde{S}^{(+)}$ and $\widetilde{S}^{(-)}$.

Size. The size of a concept indicates its generality: Small concepts (like Granny Smith) tend to be more specific than larger concepts (e.g., apple). In [8], we derived the following formula for the size M of a concept $\widetilde{S}$:[3]

$$M(\widetilde{S}) = \sum_{l=1}^{m} \left((-1)^{l+1} \cdot \sum_{\substack{\{i_1,\ldots,i_l\} \\ \subseteq \{1,\ldots,m\}}} M\left(\bigcap_{i \in \{i_1,\ldots,i_l\}} \widetilde{C}_i \right) \right)$$

with $M(\widetilde{C}) = \dfrac{\mu_0}{c^n \prod_{d \in D} w_{\delta(d)} \sqrt{w_d}} \sum_{i=0}^{n} \left(\sum_{\substack{\{d_1,\ldots,d_i\} \\ \subseteq D}} \left(\prod_{\substack{d \in \\ D \setminus \{d_1,\ldots,d_i\}}} a_d \right) \cdot \right.$

$$\left. \prod_{\substack{\delta \in \\ \Delta_{\{d_1,\ldots,d_i\}}}} \left(n_\delta! \cdot \dfrac{\pi^{\frac{n_\delta}{2}}}{\Gamma(\frac{n_\delta}{2}+1)} \right) \right)$$

Subsethood. The notion of subsethood gives rise to a concept hierarchy in a conceptual space: Because the region describing Granny Smith is a subset of the region describing apple, we know that Granny Smith is a specialization of the apple concept. We have defined a degree of subsethood as follows:

$$Sub(\widetilde{S}_1, \widetilde{S}_2) := \dfrac{M(\widetilde{S}_1 \cap \widetilde{S}_2)}{M(\widetilde{S}_1)}$$

---

[2] A strict inequality in the definition of $C^{(+)}$ or $C^{(-)}$ would not yield a cuboid.

[3] Where m is the number of cuboids in S, $\delta(d)$ is the domain to which dimension d belongs, $a_d := c \cdot w_{\delta(d)} \cdot \sqrt{w_d} \cdot (p_d^+ - p_d^-)$, $\Delta_{\{d_1,\ldots,d_i\}}$ is the domain structure that remains after removing from Δ all dimensions $d \notin \{d_1,\ldots,d_i\}$, and $n_\delta := |\delta|$.

As $\widetilde{S}_2$ sets the context for this subsethood judgement, we use $c^{(2)}$ and $W^{(2)}$ when computing the size of both the numerator and the denominator.

Implication. In order to support reasoning processes, an implication operation between different concepts is crucial. As it makes intuitive sense to consider $apple \Rightarrow red$ to be true to the degree to which $apple$ is a subset of red, we have defined the implication as follows:

$$Impl(\widetilde{S}_1, \widetilde{S}_2) := Sub(\widetilde{S}_1, \widetilde{S}_2)$$

Betweenness. We define the betweenness relation between concepts as the betweenness relation of the midpoints of their cores' central regions P.

Similarity. We define the similarity relation of two concepts as the similarity relation of the midpoints of their cores' central regions P. The sensitivity parameter c and the weights W of the second concept (which sets the context of the comparison) are used to compute this similarity value.

3 Implementation

We have implemented our formalization in Python 2.7 and have made it publicly avaliable on GitHub[4] [6]. Figure 3 shows a class diagram illustrating the overall structure of our implementation. As one can see, each of the components from our definition (i.e., weights, cuboids, cores, and concepts) is represented by an individual class. Moreover, the "cs" module contains the overall domain structure of the conceptual space (represented as a dictionary mapping from domain identifiers to sets of dimensions) along with some utility functions (e.g., computing distance and betweenness of points). In order to define a new concept, one needs to use all of the classes, as all components of the concept need to be specified in detail. When operating with the concepts, it is however sufficient to use the `Concept` class which contains all the operations defined in Section 2.3.

The implementation of most operations is rather straightforward. For instance, the subspace projection can be implemented by removing domains both from all cuboids of a concept and from its weights. The details about how these operations have been implemented are thus omitted from this paper. The intersection operation, however, has a more complex implementation and will be discussed in more detail.

The key challenge with respect to the intersection is to find the new core S', i.e. the highest non-empty α-cut intersection of the two sets. We simplify this problem by iterating over all combinations of cuboids $C_1 \in S_1, C_2 \in S_2$ and by looking at each pair of cuboids individually. In order to find the highest
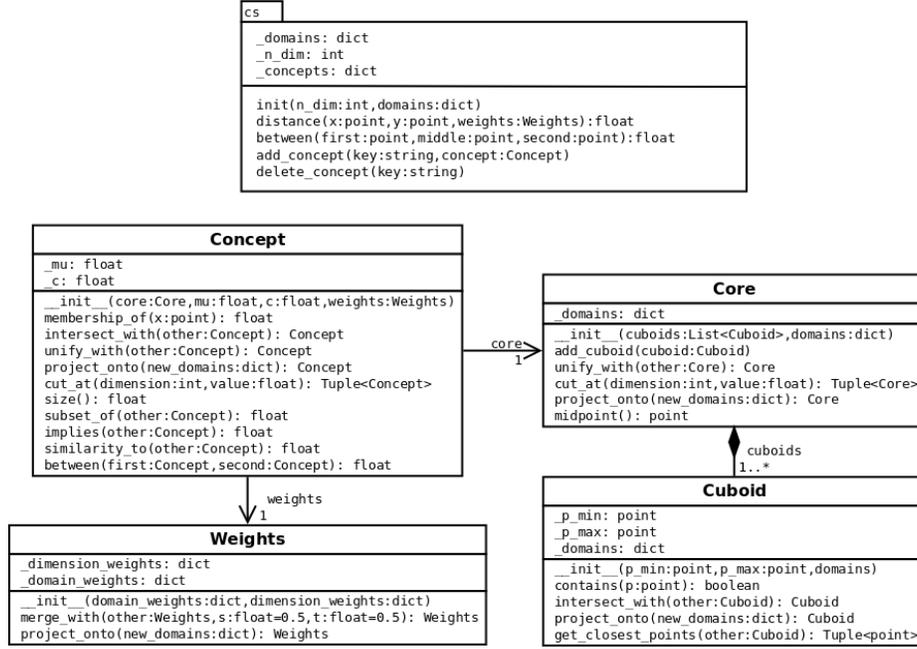
---

[4] See `https://github.com/lbechberger/ConceptualSpaces/tree/v1.0.0`

```
cs
_domains: dict
_n_dim: int
_concepts: dict

init(n_dim:int,domains:dict)
distance(x:point,y:point,weights:Weights):float
between(first:point,middle:point,second:point):float
add_concept(key:string,concept:Concept)
delete_concept(key:string)
```

```
Concept
_mu: float
_c: float

__init__(core:Core,mu:float,c:float,weights:Weights)
membership_of(x:point): float
intersect_with(other:Concept): Concept
unify_with(other:Concept): Concept
project_onto(new_domains:dict): Concept
cut_at(dimension:int,value:float): Tuple<Concept>
size(): float
subset_of(other:Concept): float
implies(other:Concept): float
similarity_to(other:Concept): float
between(first:Concept,second:Concept): float
```

```
Core
_domains: dict

__init__(cuboids:List<Cuboid>,domains:dict)
add_cuboid(cuboid:Cuboid)
unify_with(other:Core): Core
cut_at(dimension:int,value:float): Tuple<Core>
project_onto(new_domains:dict): Core
midpoint(): point
```

```
Weights
_dimension_weights: dict
_domain_weights: dict

__init__(domain_weights:dict,dimension_weights:dict)
merge_with(other:Weights,s:float=0.5,t:float=0.5): Weights
project_onto(new_domains:dict): Weights
```

```
Cuboid
_p_min: point
_p_max: point
_domains: dict

__init__(p_min:point,p_max:point,domains)
contains(p:point): boolean
intersect_with(other:Cuboid): Cuboid
project_onto(new_domains:dict): Cuboid
get_closest_points(other:Cuboid): Tuple<point>
```

Fig. 3. Class diagram of our implementation.

α-cut intersection of the two cores, we simply take the maximum over all pairs of cuboids. Let $a \in C_1$ and $b \in C_2$ be the two closest points from the two cuboids under consideration (i.e., $\forall x \in C_1, y \in C_2 : d(a,b) \leq d(x,y)$). When intersecting two fuzzified cuboids[5] $\widetilde{C}_1$ and $\widetilde{C}_2$, the following results are possible:

1. The crisp cuboids have a nonempty intersection (Figure 4a). In this case, we simply compute their crisp intersection.
2. The μ_0 parameters are different and the $\mu_0^{(i)}$-cut of $\widetilde{C}_j$ intersects with C_i (Figure 4b). In this case, we need to intersect $\widetilde{C}_j^{\mu_0^{(i)}}$ with C_i and approximate the result by a cuboid.
3. The intersection of the two fuzzified cuboids consists of a single point x^* lying between a and b (Figure 4c). In this case, we define a trivial cuboid with $p^- = p^+ = x^*$.
4. The intersection of the two fuzzified cuboids consists of a set of points (Figure 4d). This can only happen if the α-cut boundaries of both fuzzified cuboids are parallel to each other, which requires multiple domains to be involved and the weights of both concepts to be linearly dependent.

Algorithm 1 shows how the intersection is implemented. Lines 2 & 3 cover the crisp intersection. After finding a pair of closest points a, b, we ignore from our

---

[5] The membership function $\mu_{\widetilde{C}}(x)$ can be obtained by replacing S with C in Def. 4.

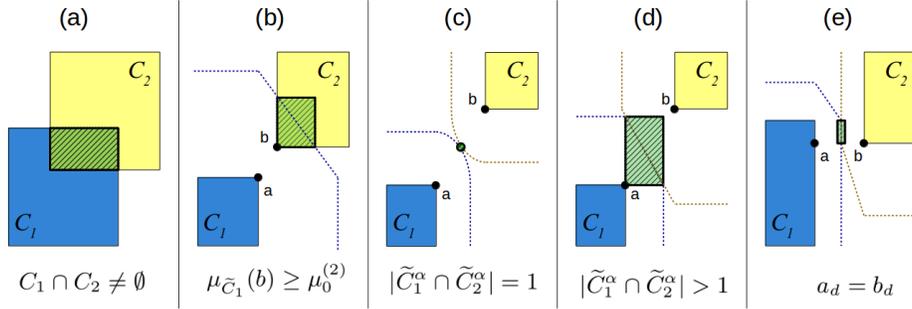| (a)	(b)	(c)	(d)	(e)
$C_1 \cap C_2 \neq \emptyset$	$\mu_{\widetilde{C}_1}(b) \geq \mu_0^{(2)}$	$|\widetilde{C}_1^\alpha \cap \widetilde{C}_2^\alpha| = 1$	$|\widetilde{C}_1^\alpha \cap \widetilde{C}_2^\alpha| > 1$	$a_d = b_d$

Fig. 4. Possible results of intersecting two fuzzy cuboids.

further considerations all dimensions where $a_d = b_d$ (lines 5 & 6). Lines 7-10 cover the second case. We use numerical optimization (from the `scipy.optimize` package) to find x^* in line 12. If the weights are linearly dependent (line 14), we deal with the fourth case from above (lines 15-22): We look for points on the surface of the bounding box spanned by the points a and b that are both in $\widetilde{C}_1^\alpha$ and $\widetilde{C}_2^\alpha$. We iteratively look at the edges ($i = 1$), faces ($i = 2$), etc. of the bounding box until we find such points. We then approximate them with a cuboid. If the weights are not linearly dependent, we are in case 3 from above (line 24). Finally, in line 27 we extrude the identified cuboid in all dimensions where $a_d = b_d$ and where a_d and b_d can vary (cf. Figure 4e).

4 Example

Our implementation of the conceptual spaces framework contains a simple toy example – a three-dimensional conceptual space for fruits, defined as follows:

$$\Delta = \{\delta_{color} = \{d_{hue}\}, \delta_{shape} = \{d_{round}\}, \delta_{taste} = \{d_{sweet}\}\}$$

d_{hue} describes the hue of the observation's color, ranging from 0.00 (purple) to 1.00 (red). d_{round} measures the percentage to which the bounding circle of an object is filled. d_{sweet} represents the relative amount of sugar contained in the fruit, ranging from 0.00 (no sugar) to 1.00 (high sugar content). As all domains are one-dimensional, the dimension weights w_d are always equal to 1.00 for all concepts. We assume that the dimensions are ordered like this: $d_{hue}, d_{round}, d_{sweet}$. Table 1 defines some concepts in this space[6] and Figure 5 visualizes them. The conceptual space is defined as follows in the code:

```
domains = {'color':[0], 'shape':[1], 'taste':[2]}
space.init(3, domains)
```

Concepts can be defined as follows:

---

[6] Due to space restrictions, we only show a subset of the concepts defined in the demo.

Algorithm 1 Highest non-empty α-cut of two cuboids

1: **function** INTERSECT_CUBOIDS(Cuboid C_1, Cuboid C_2, Concept $\widetilde{S}_1$, Concept $\widetilde{S}_2$)
2:     **if** $C_1 \cap C_2 \neq \emptyset$ **then**
3:         $\alpha \leftarrow \min(\mu_0^{(1)}, \mu_0^{(2)})$, $C \leftarrow C_1 \cap C_2$
4:     **else**
5:         Find closest points $a \in C_1, b \in C_2$
6:         Ignore all dimensions d where $a_d = b_d$
7:         **if** $\mu_{\widetilde{C}_1}(b) \geq \mu_0^{(2)}$ **then**
8:             $\alpha \leftarrow \mu_0^{(2)}$, $C \leftarrow$ cuboid approximation of $\widetilde{C}_1^{\mu_0^{(2)}} \cap C_2$
9:         **else if** $\mu_{\widetilde{C}_2}(a) \geq \mu_0^{(1)}$ **then**
10:            $\alpha \leftarrow \mu_0^{(1)}$, $C \leftarrow$ cuboid approximation of $\widetilde{C}_2^{\mu_0^{(1)}} \cap C_1$
11:         **else**
12:            Find $x^* = \arg\max_{x \in CS}(\mu_{\widetilde{C}_1}(x))$ with $\mu_{\widetilde{C}_1}(x) = \mu_{\widetilde{C}_2}(x)$
13:            $\alpha \leftarrow \mu_{\widetilde{C}_1}(x^*)$
14:            **if** $\exists t \in \mathbb{R} : \forall d$ with $a_d \neq b_d : w_{\delta(d)}^{(1)} \cdot \sqrt{w_d^{(1)}} = t \cdot w_{\delta(d)}^{(2)} \cdot \sqrt{w_d^{(2)}}$ **then**
15:                **for** $i = 1$ to $|\{d : a_d \neq b_d\}| - 1$ **do**
16:                    Find all x on the i-faces of the bounding box spanned by a and b
17:                    with $\mu_{\widetilde{C}_1}(x) = \mu_{\widetilde{C}_2}(x) = \alpha$
18:                  **if** found at least one x **then**
19:                    $C \leftarrow$ cuboid-approximation of the set of all x
20:                    **break**
21:                  **end if**
22:                **end for**
23:              **else**
24:                 $C \leftarrow$ trivial cuboid consisting of x^*
25:              **end if**
26:         **end if**
27:         Extrude C in all dimensions d where $a_d = b_d$
28:     **end if**
29:     **return** α, C
30: **end function**

```
c_pear = Cuboid([0.5, 0.4, 0.35], [0.7, 0.6, 0.45], domains)
s_pear = Core([c_pear], domains)
w_pear = Weights({'color':0.50, 'shape':1.25, 'taste':1.25}, {'color
    ':{0:1.0}, 'shape':{1:1.0}, 'taste':{2:1.0}})
pear = Concept(s_pear, 1.0, 12.0, w_pear)
```

We can load the definition of this fruit space into our python interpreter and apply the different operations described in Section 2 to these concepts. This looks for example as follows:

```
>>> execfile('fruit_space.py')
>>> granny_smith.subset_of(apple)
1.0
>>> apple.implies(red)
0.3333333333333332
>>> apple.between(lemon, orange)
```

Concept	Δ_S	p^-	p^+	μ_0	c	W		
						$w_{\delta_{color}}$	$w_{\delta_{shape}}$	$w_{\delta_{taste}}$
Pear	Δ	(0.50, 0.40, 0.35)	(0.70, 0.60, 0.45)	1.0	12.0	0.50	1.25	1.25
Orange	Δ	(0.80, 0.90, 0.60)	(0.90, 1.00, 0.70)	1.0	15.0	1.00	1.00	1.00
Lemon	Δ	(0.70, 0.45, 0.00)	(0.80, 0.55, 0.10)	1.0	20.0	0.50	0.50	2.00
Granny Smith	Δ	(0.55, 0.70, 0.35)	(0.60, 0.80, 0.45)	1.0	25.0	1.00	1.00	1.00
Apple	Δ	(0.50, 0.65, 0.35) (0.65, 0.65, 0.40) (0.70, 0.65, 0.45)	(0.80, 0.80, 0.50) (0.85, 0.80, 0.55) (1.00, 0.80, 0.60)	1.0	10.0	0.50	1.50	1.00
Red	$\{\delta_{color}\}$	(0.90, -∞, -∞)	(1.00, +∞, +∞)	1.0	20.0	1.00	–	–

Table 1. Definitions of several concepts.

```
1.0
>>> pear.similarity_to(apple)
0.007635094218859955
>>> pear.similarity_to(lemon)
1.8553913626159717e-07
>>> print apple.intersect_with(pear)
core: {[0.5, 0.625, 0.35]-[0.7, 0.625, 0.45]}
mu: 0.6872892788
c: 10.0
weights: <{'color': 0.5, 'taste': 1.125, 'shape': 1.375},
    {'color': {0: 1.0}, 'taste': {2: 1.0}, 'shape': {1: 1.0}}>
>>> apple.size(), pear.size()
(0.10483333333333335, 0.041481481481481466)
>>> (apple.unify_with(pear)).size()
0.146900844381
>>> print lemon.project_onto({"color":[0]})
core: {[0.7, -inf, -inf]-[0.8, inf, inf]}
mu: 1.0
c: 20.0
weights: <{'color': 1.0},{'color': {0: 1.0}}>
```

5 Related Work

Our work is of course not the first attempt to devise an implementable formalization of the conceptual spaces framework.

An early and very thorough formalization was done by Aisbett & Gibbon [4]. Like we, they consider concepts to be regions in the overall conceptual space. However, they stick with Gärdenfors' assumption of convexity and do not define concepts in a parametric way. Their formalization targets the interplay of symbols and geometric representations, but it is too abstract to be implementable.

Rickard [23] provides a formalization based on fuzziness. He represents concepts as co-occurence matrices of their properties. By using some mathematical transformations, he interprets these matrices as fuzzy sets on the universe of
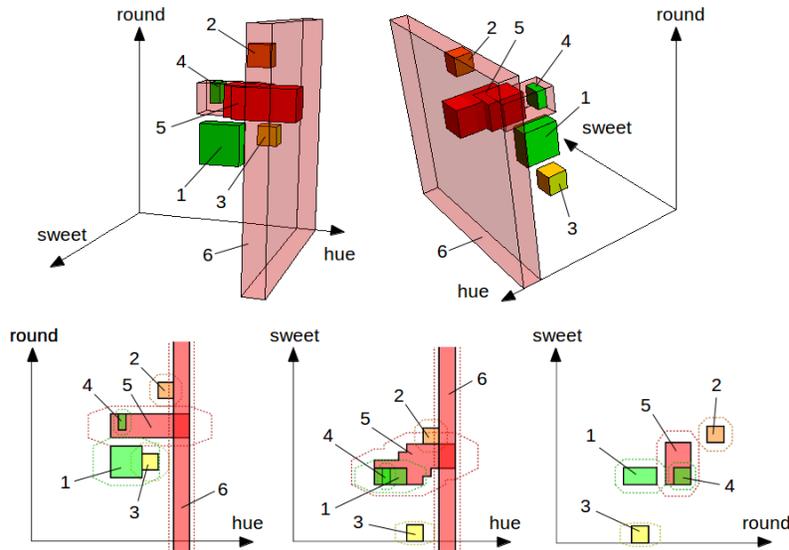
Fig. 5. Top: Three-dimensional fruit space (only cores). Bottom: Two-dimensional visualization of the fruit space (cores and 0.5-cuts). The concepts are labeled as follows: pear (1), orange (2), lemon (3), Granny Smith (4), apple (5), red (6).

ordered property pairs. As properties and concepts are represented in different ways, one has to use different learning and reasoning mechanisms for them. Rickard's formalization is also not easy to work with due to the complex mathematical transformations involved.

Adams & Raubal [1] represent concepts by one convex polytope per domain. This allows for efficient computations while being potentially more expressive than our cuboid-based representation. However, correlations between different domains are not taken into account. Adams & Raubal also define operations on concepts, namely intersection, similarity computation, and concept combination. This makes their formalization quite similar in spirit to ours.

Lewis & Lawry [20] formalize conceptual spaces using random set theory. They define properties as random sets within single domains, and concepts as random sets in a boolean space whose dimensions indicate the presence or absence of properties. Their approach is similar to ours in using a distance-based membership function to a set of prototypical points. However, their work purely focuses on modeling conjunctive concept combinations and does not consider correlations between domains.

To the best of our knowledge, none of the above mentioned formalizations have a publicly accessible implementation. In [21], Lieto et al. present a hybrid architecture that represents concepts by using both description logics and conceptual spaces. This way, symbolic ontological information and similarity-based "common sense" knowledge can be used in an integrated way. Each concept is

represented in the conceptual space by a single prototypical point and a number of exemplar points. Correlations between domains can therefore only be encoded through the selection of appropriate exemplars. Their work focuses on classification tasks and does therefore not provide any operations for combining different concepts. With respect to the larger number of supported operations, our implementation can therefore be considered more general than theirs. The implementation of their system[7] is the only publicly available implementation of the conceptual spaces framework we are currently aware of. In contrast to our work, it however comes without any publicly available source code[8].

6 Conclusion and Future Work

In this paper, we presented a comprehensive implementation of the conceptual spaces framework. This implementation and its source code are publicly avaliable and can be used by any researcher interested in conceptual spaces. We think that our implementation can be a good foundation for practical research on conceptual spaces and that it will considerably facilitate research in this area.

In future work, we will implement a visualization toolbox in order to enrich the presented implementation with visual output. Moreover, we will use this implementation to apply machine learning algorithms in conceptual spaces. Needless to say, any future extensions of our formalization will also be incorporated into future versions of this implementation.

References

1. Adams, B., Raubal, M.: A Metric Conceptual Space Algebra. In: 9th International Conference on Spatial Information Theory. pp. 51–68. Springer Berlin Heidelberg (2009)
2. Adams, B., Raubal, M.: Conceptual Space Markup Language (CSML): Towards the Cognitive Semantic Web. 2009 IEEE International Conference on Semantic Computing (Sep 2009)
3. Aggarwal, C.C., Hinneburg, A., Keim, D.A.: On the Surprising Behavior of Distance Metrics in High Dimensional Space. In: 8th International Conference on Database Theory. pp. 420–434. Springer Berlin Heidelberg (2001)
4. Aisbett, J., Gibbon, G.: A General Formulation of Conceptual Spaces as a Meso Level Representation. Artificial Intelligence 133(1-2), 189–232 (Dec 2001)
5. Attneave, F.: Dimensions of Similarity. The American Journal of Psychology 63(4), 516–556 (Oct 1950)
6. Bechberger, L.: lbechberger/ConceptualSpaces: Version 1.0.0. `https://doi.org/10.5281/zenodo.825459` (Jul 2017)
7. Bechberger, L., Kühnberger, K.U.: A Thorough Formalization of Conceptual Spaces. In: Kern-Isberner, G., Fürnkranz, J., Thimm, M. (eds.) KI 2017: Advances

---

[7] See `http://www.dualpeccs.di.unito.it/download.html`.

[8] The source code of an earlier and more limited version of their system can be found here: `http://www.di.unito.it/~lieto/cc_classifier.html`.

in Artificial Intelligence: 40th Annual German Conference on AI, Dortmund, Germany, September 25–29, 2017, Proceedings. pp. 58–71. Springer International Publishing (2017), `https://doi.org/10.1007/978-3-319-67190-1_5`

8. Bechberger, L., Kühnberger, K.U.: Measuring Relations Between Concepts In Conceptual Spaces. In: Thirty-seventh SGAI International Conference on Artificial Intelligence (in press), `https://arxiv.org/abs/1707.02292`

9. Bělohlávek, R., Klir, G.J.: Concepts and Fuzzy Logic. MIT Press (2011)

10. Billman, D., Knutson, J.: Unsupervised Concept Learning and Value Systematicitiy: A Complex Whole Aids Learning the Parts. Journal of Experimental Psychology: Learning, Memory, and Cognition 22(2), 458–475 (1996)

11. Chella, A., Frixione, M., Gaglio, S.: Conceptual Spaces for Computer Vision Representations. Artificial Intelligence Review 16(2), 137–152 (2001)

12. Chella, A., Frixione, M., Gaglio, S.: Anchoring Symbols to Conceptual Spaces: The Case of Dynamic Scenarios. Robotics and Autonomous Systems 43(2-3), 175–188 (May 2003)

13. Derrac, J., Schockaert, S.: Inducing Semantic Relations from Conceptual Spaces: A Data-Driven Approach to Plausible Reasoning. Artificial Intelligence 228, 66–94 (Nov 2015)

14. Douven, I., Decock, L., Dietz, R., Égré, P.: Vagueness: A Conceptual Spaces Approach. Journal of Philosophical Logic 42(1), 137–160 (Nov 2011)

15. Fiorini, S.R., Gärdenfors, P., Abel, M.: Representing Part-Whole Relations in Conceptual Spaces. Cognitive Processing 15(2), 127–142 (Oct 2013)

16. Gärdenfors, P.: Conceptual Spaces: The Geometry of Thought. MIT press (2000)

17. Gärdenfors, P.: The Geometry of Meaning: Semantics Based on Conceptual Spaces. MIT Press (2014)

18. Harnad, S.: The Symbol Grounding Problem. Physica D: Nonlinear Phenomena 42(1-3), 335–346 (Jun 1990)

19. Hernández-Conde, J.V.: A Case Against Convexity in Conceptual Spaces. Synthese 193, 1–27 (May 2016)

20. Lewis, M., Lawry, J.: Hierarchical Conceptual Spaces for Concept Combination. Artificial Intelligence 237, 204–227 (Aug 2016)

21. Lieto, A., Radicioni, D.P., Rho, V.: Dual PECCS: a cognitive system for conceptual representation and categorization. Journal of Experimental & Theoretical Artificial Intelligence 29(2), 433–452 (2017), `http://dx.doi.org/10.1080/0952813X.2016.1198934`

22. Murphy, G.: The Big Book of Concepts. MIT Press (2002)

23. Rickard, J.T.: A Concept Geometry for Conceptual Spaces. Fuzzy Optimization and Decision Making 5(4), 311–329 (Oct 2006)

24. Schockaert, S., Prade, H.: Interpolation and Extrapolation in Conceptual Spaces: A Case Study in the Music Domain. In: 5th International Conference on Web Reasoning and Rule Systems. pp. 217–231. Springer Nature (2011)

25. Shepard, R.N.: Attention and the Metric Structure of the Stimulus Space. Journal of Mathematical Psychology 1(1), 54–87 (Jan 1964)

26. Smith, C.R.: A Characterization of Star-Shaped Sets. The American Mathematical Monthly 75(4), 386 (Apr 1968)

27. Warglien, M., Gärdenfors, P., Westera, M.: Event Structure, Conceptual Spaces and the Semantics of Verbs. Theoretical Linguistics 38(3-4), 159–193 (Jan 2012)

28. Zadeh, L.A.: Fuzzy Sets. Information and Control 8(3), 338–353 (Jun 1965)

29. Zadeh, L.A.: A Note on Prototype Theory and Fuzzy Sets. Cognition 12(3), 291–297 (Jan 1982)