In *Proceedings of the AAAI Fall Symposium on Advances in Cognitive Systems, 2011.*

Bridging Dichotomies in Cognitive Architectures for Virtual Humans

Paul S. Rosenbloom
Department of Computer Science & Institute for Creative Technologies
University of Southern California
12015 Waterfront Drive, Playa Vista, CA 90094
rosenbloom@usc.edu

Abstract

Desiderata for cognitive architectures that are to support the extent of human-level intelligence required in virtual humans imply the need to bridge a range of dichotomies faced by such architectures. The focus here is first on two general approaches to building such bridges – *addition* and *reduction* – and then on a pair of general tools – *graphical models* and *piecewise continuous functions* – that exploit the second approach towards developing such an architecture. Evaluation is in terms of the architecture's demonstrated ability and future potential for bridging the dichotomies.

Introduction

Virtual humans combine human-like cognition with virtual human bodies in interactive games and simulations for education, training and entertainment. The concern here is how to build cognitive architectures for such virtual humans that are: (1) *broad spectrum*, (2) *tightly integrated*, and (3) *functionally elegant*. A broad-spectrum architecture makes creating simple virtual humans simple, often by leveraging statistical techniques over large data sets, while still enabling construction of the most sophisticated ones – capable of symbolic reasoning over complex models – and being incrementally extendable to points in between. A tightly integrated architecture closely couples capabilities both within the central cognitive system and between it and the perceptuomotor system, enabling effective sharing of information and uncertainty across capabilities plus the creation of capability combinations that are much more than just the sum of their parts. A functionally elegant architecture yields systems with the broad functionality implicated by human-level intelligence while remaining theoretically simple, maintainable and extendible.

When considering the related goal of achieving human-level intelligence in cognitive systems, the same desiderata still matter, but the arguments are not always identical, and there is less agreement about some aspects. Tight integration remains just as critical. With respect to broad spectrum, the ability to construct simple statistical systems may no longer be essential, but combining statistical and symbolic techniques is still likely important. While functionality remains central, there is little consensus as to whether it is either possible or desirable to combine it with simplicity and elegance. However, earlier work on the *diversity dilemma* – of how to combine broad functionality, which usually implies a large number of specialized architectural mechanisms, with simplicity and elegance, which typically implies a small number of general mechanisms – does explore the hypothesis that functional elegance is possible, while simultaneously arguing for its desirability (Rosenbloom 2011a).

While the diversity dilemma can be seen as a variant of functional elegance, it can also be seen as just one dichotomy – uniform vs. diverse – out of many that must be resolved to enable cognitive architectures that satisfy the full set of desiderata. A broad-spectrum architecture, for example, may require bridging the logical vs. probabilistic and data-driven vs. model-based dichotomies. Tight integration may require bridging such dichotomies as procedural vs. declarative, reactive vs. deliberative, discrete vs. continuous and central vs. peripheral. Other dichotomies that may also need bridging include symbolic vs. subsymbolic/neural, explicit vs. implicit, hard vs. soft constraints and goal vs. utility based. These dichotomies are not all necessarily independent of each other, with some mapped onto others by particular theoretical stances, but there are discernable differences among all of them.

Individual architectures all take some stance on each such dichotomy, either opting for one side or the other – based on theoretical and/or pragmatic reasons – or bridging the dichotomy to provide both in some form. Choosing one alternative is generally the simplest approach, and it can often work, at least until the challenges faced by the architecture become too diverse. Yet it is inadequate for most dichotomies over the long run, particularly given the need for virtual humans to approach human-level intelligence in a broad sociophysical context. Here the question of how to bridge the dichotomies must be faced.

This article begins with two general approaches for bridging such dichotomies – *addition* and *reduction* – and then discusses two general tools – *graphical models* (Koller and Friedman 2009) and *piecewise continuous functions* – that exploit the reduction approach towards developing a *graphical architecture* that bridges many of the listed dichotomies. Graphical models were central to the earlier work on the diversity dilemma, but will be treated in an abbreviated manner here. Instead, piecewise continuous functions – which have heretofore been lurking in the background – are brought to the fore as a major idea in their own right. It is these piecewise functions that define the elementary cognitive structures in the architecture, yielding a representational primitive that is key in bridging many dichotomies. The graphical models then specify how these basic units are combined into complex mental structures and processed, completing the bridges and yielding sophisticated cognitive capabilities. Results to date, in conjunction with future prospects already within view, demonstrate how these tools leverage reduction to bridge many of the dichotomies central to developing architectures for virtual humans (and human-level cognitive systems).

Two Approaches to Bridging Dichotomies

The most obvious approach to bridging a dichotomy is *addition* of a module for each alternative. ACT-R, for example, includes distinct procedural and declarative memories (Anderson et al. 2004); and Soar 9 goes further, reifying a subsidiary dichotomy between semantic and episodic knowledge (Laird 2008). Clarion also embodies distinct procedural and declarative systems, but with both explicit and implicit implementations of each (Sun 2006). The 3T architecture includes modules for reactive and deliberative control (Bonasso et al. 1997). EPIC has a central cognitive module plus multiple peripheral modules (Kieras and Meyer 1997). Addition can broadly contribute to the functional side of functional elegance, while yielding multiple points on a spectrum of capabilities. It doesn't, however, yield either a full capability spectrum or elegance; at worst producing a combinatoric proliferation of mechanisms, when bridging N dichotomies engenders 2^N rather than $2N$ components (as with Clarion for the two dichotomies mentioned). Addition is neutral in general about how to achieve tight integration.

Reduction instead leverages whatever commonality exists across a dichotomy to bridge it via a single module that is little more complex than what is necessary for just one side. In the purest cases, the commonality captures all that matters of both sides of the dichotomy. However, there may be residual differences that require further attention or a functional compromise may be required on one or both sides to maximize the commonality. Whether or not such additional complexities do occur, there are at least three distinct forms of commonality that can be identified: (1) a generalization that directly subsumes both sides of the dichotomy; (2) one side of the dichotomy implementing the other; or (3) a generalization beneath the dichotomy that implements both sides of it.

Markov logic (Domingos and Lowd 2009) provides a canonical example of the first form of reduction, with its generalization over first order logic and probabilities. It also bridges other related dichotomies as well; such as data-driven vs. model-based, where statistical learning over large bodies of data provides the robust breadth characteristic of data-driven processing and first-order logic provides the robust depth characteristic of model-based processing. Rule-based architectures, at least as applied in classical shallow expert systems, can also be seen as a subsumption-based generalization reduction of the data-driven vs. model-based dichotomy, with breadth stemming from large numbers of shallow rules and depth from the first-order generalizations yielded by variables and combinatoric match. However, this is an attenuated generalization that imposes compromises on both sides of the dichotomy. It sacrifices both breadth and depth with respect to the pure alternatives, while requiring a double sacrifice on robustness – the lack of both statistics over large datasets and combinatoric search over comprehensive models leads to the classic problem of rule fragility. What is gained in return is efficiency (and simplicity).

Work on Soar has provided several examples of the second form of reduction, some positive and others not. On the positive side, to bridge reactivity vs. deliberation, Soar's reactive component – parallel rules – provides an associative memory that implements much of deliberation (Laird and Rosenbloom 1990). A decision procedure is also needed for deliberation, but this is a minor addition in comparison. More problematically, bridging the procedural vs. declarative dichotomy by implementing a declarative capability on top of the architecture's procedural learning (chunking) and memory (rules) yielded an awkward result that integrated poorly with the rest of the system (Rosenbloom, Laird and Newell 1987). Ultimately this led to adopting the addition approach for this dichotomy in Soar 9. In general, introducing a level difference between the two sides of a dichotomy – with the lower implementing the upper – implies an order-of magnitude difference in time scale, and integration that is not among peers. When this works, as for reactivity vs. deliberation, this form of reduction can be effective. When it doesn't, the result can be awkward and poorly integrated.

The graphical architecture illustrates the third form of reduction. For example, a general *implementation level* beneath the architecture – based on factor graphs and the summary product algorithm (Kschischang, Frey and Loeliger 2001) – bridged the procedural vs. declarative dichotomy by implementing both kinds of memories, with just three residual differences: whether message passing was unidirectional vs. bidirectional, whether working memory was closed vs. open world, and whether variable binding concerned all legal values vs. only the single best value (Rosenbloom 2010). Choices concerning these differences could all be made independently, engendering a larger space of possible memories, including a constraint memory that blends procedural and declarative aspects.

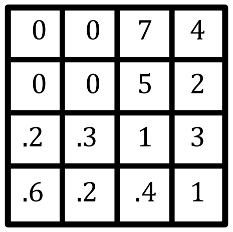
By uncovering and exploiting a deeper commonality between the alternatives, the reduction approach can yield: simpler systems, proffering the elegance side of functional elegance; insight into how to integrate the alternatives, through what is shared; and deep scientific results that are of interest in their own right. When reduction is based on generalization, it is also possible to go beyond just bridging a pair of alternatives, to yield a full spectrum of options.

Piecewise Continuous Functions

Piecewise continuous functions are simple in their basic conception, but complex in their full implications. Consider first a general form for arbitrary (non-piecewise) multivariate functions: $f(x_1, \ldots, x_n)$. This provides an extreme variant of the first form of reduction, subsuming many varieties of basic representational units. A Boolean function of two discrete variables yields, for example, a template for 2D occupancy grids. A real-valued function of three continuous variables yields a template for state functions over the spatial dimensions of the physical universe: $f(x, y, z)$. Addition of a fourth continuous variable (t) enables representing the dynamics of such states over time. On the other hand, a Boolean function – where *false* and *true* map onto values of 0 and 1 – over symbolic variables yields a template for logical predicates and n-ary relations – such as *Color*(*object*, *hue)* – while real-valued functions restricted to the closed interval [0, 1] yield a template for probability distributions, such as $P(x, y)$ and $P(x \mid y)$, when defined over discrete variables.

Other kinds of functions, such as utility functions, can also be represented in comparable ways, but these examples should be sufficient to make the case. The overall qualitative space of variation covers the number of independent variables (N), the nature of the domains of the individual variables, and the restrictions imposed on the range of the function. Within each point in this qualitative space, many distinct functions can then be defined in support of many different representations. The problem though with building architectures around such a representation is that specifying and computing with them can be all over the map, as well as arbitrarily difficult. In the graphical architecture, it is necessary to compute: (1) whether two functions are equal; (2) the pointwise product or sum of two functions; (3) the effect of integrating or maximizing over a variable; and (4) changes in a function's value over an arbitrary segment of its domain.

Piecewise continuous functions provide an alternative generalization that compromises on expressibility in exchange for radical improvements in simplicity and efficiency. The essence is to view a function's variables as defining a multidimensional continuous space, with one dimension per variable, and then to decompose this space into regions, each with a simpler function defined on it. Regionalization introduces an aspect of discreteness into the functional form, while enabling a trade off – by varying the sizes of the regions – between the degree to which an arbitrary continuous function can be approximated and the cost of representing and processing it.



0	0	7	4
0	0	5	2
.2	.3	1	3
.6	.2	.4	1

Figure 1: Piecewise constant function over 2D hypercubes (squares).

The simplest variety of piecewise continuous functions limits regions to *hypercubes* – N dimensional generalizations of squares and cubes – of uniform size, and restricts region functions to be constant (Figure 1). In one dimension this yields a step function whose value can change at fixed intervals. In two dimensions it supports pixels and in three dimensions we get voxels. Piecewise constant functions also subsume both logical predicates and discrete probability distributions. For predicates, symbols are mapped onto unit intervals along individual dimensions and constant values are limited to 0/1 (for False/True) over combinations of these dimensions. For distributions, unit intervals are assigned constant values in [0, 1]. Although some overhead is introduced in representing predicates and discrete distributions in this uniform manner, as long as it remains a small constant factor it should not be an issue.

Representation via piecewise constant functions can directly support limited forms of continuous functions – step functions – while approximating arbitrary ones as closely as desired; although the size of the representation for continuous functions grows by 2^N as the interval widths are halved. Still, the important point is that this subsumption-based generalization reduction provides the same representational interface for continuous functions as for logical predicates and discrete probability distributions, opening up the possibility – which can be exploited by graphical models – of providing uniform processing algorithms over these disparate forms of data.

The basic representational aspects of both sides of many of the dichotomies mentioned in the introduction can be handled by these simple piecewise functions. The uniform vs. diverse dichotomy is handled by the overall generalization that enables a diversity of elementary representational units to all be subsumed by a single representation. Boolean function ranges being a special case of real ones handles the logical vs. probabilistic dichotomy. The representational aspects of many additional dichotomies are then handled by how this generalization covers both symbolic/relational and continuous data. This bridges in particular much of the explicit vs. implicit, symbolic vs. subsymbolic, hard vs. soft constraints, central vs. peripheral, discrete vs. continuous, and goal-based vs. utility-based dichotomies.

To make this more concrete, we'll examine how the graphical architecture currently exploits a slightly more sophisticated form of piecewise continuous function to address four rather different representational challenges found in cognitive architectures: (1) working memory, (2) mental imagery, (3) probability distributions, and (4) episodic memory. We'll then look at potential enhancements for strengthening, in particular, the bridges between discrete vs. continuous, symbolic vs. subsymbolic and central vs. peripheral.

The graphical architecture extends the piecewise constant hypercube representation to piecewise linear functions over *hyperrectangles*, or *orthotopes* (Figure 2). Linear region functions enable exact representation of a broader range of continuous functions plus more compact approximations of many others. They do introduce additional complexity, including the need for reapproximation in cases where computations over linear functions yield nonlinear functions – such as when the product of two linear functions yields a quadratic function – yet the computations remain relatively simple overall.

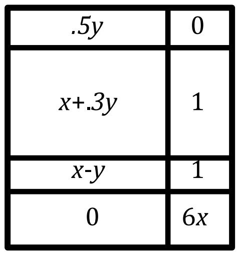


Figure 2: Array of piecewise linear functions over axially aligned 2D orthotopes.

Shifting from an array of hypercubes to one of hyperrectangles enables region widths to vary across segments of a single dimension, as well as between dimensions, while still retaining the simplification that the borders between regions align into *slices* that span the function. Whenever an entire region no longer has the same linear function, slices are automatically added to partition the region into smaller, functionally homogeneous ones. Similarly, whenever all pairs of regions straddling a slice use the same linear function, the slice is removed to minimize the representation. Function-wide slices imply more fragmentation than is logically necessary, but in exchange they maintain the function as an array of regions, enabling function traversal (for computational purposes) to remain systematic and straightforward.

Working Memory

In a symbolic architecture such as Soar, working memory (WM) consists of object-attribute-value triples, defining a graph of symbolic relations. Using piecewise functions, the graphical architecture can represent such a WM, while simultaneously extending it for continuous information. A set of predicates is first defined, each over a fixed number of typed arguments. A variable's type determines the extent of the underlying dimension, whether the dimension is continuous or discrete (implying all slices are at integral values), and the mapping of symbols onto the discrete domain (if the dimension is symbolic). With a symbolic object-attribute-value triple – such as might encode the color of an object – the attribute (color) defines the predicate, and two symbolic arguments range over the objects and the colors: *Color*(*object*, *color*).

It is important to note that although type distinctions enable compiling a diversity of knowledge into the representation, their processing remains unaware of these differences. As far as the graphical models are concerned, all dimensions are continuous. For each predicate, a Boolean WM function with a dimension for each argument is created to represent which regions of the predicate are in WM. Active regions have a functional value of 1 while inactive regions are 0, as illustrated in Figure 3.

	Red	**Green**	**Yellow**	**Blue**
O1	0	1	0	0
O2	0	0	1	0
O3				
O4	1	0	0	0

Figure 3: Partial Boolean WM function with Color(O1,Green), Color(O2,Yellow), Color(O3,Yellow), Color(O4,Red) active.

By keeping the regions in this function as large as possible within the expressibility of the piecewise function used, significant compactness can be maintained. Even more importantly, regionalization enables an infinite number of continuous values to be represented implicitly within a region. The result is a *hybrid* WM that represents both discrete/symbolic and continuous information, and that can mix them freely across the variables/dimensions of predicates. For example, an Eight Puzzle board (Figure 4) can be defined via a hybrid *Board*(*x*, *y*, *tile)* predicate, where *x* and *y* are continuous with extent [0, 3), and *tile* is discrete with extent 0-9 (Rosenbloom 2011b). Combinations of predicates then afford richly structured networks of relations in WM.

Figure 4: Eight Puzzle board.

Mental Imagery

Mental imagery involves a multidimensional representation of space and its contents, plus the ability to transform such a representation. Piecewise continuous functions are well suited to this. One continuous dimension is required for each spatial dimension, plus an additional dimension for objects within this space. For example, two spatial dimensions can be combined with a color dimension to define a three-argument predicate capable of representing a color map over a surface. Or, three spatial dimensions can be combined with an object dimension to define a four-argument predicate that represents a 3D occupancy grid.

The Eight Puzzle board, with its two continuous spatial dimensions and one discrete tile dimension, provides a concrete example in Figure 5. When combined with the implementation of a translation operation in the graphical model, plus additional knowledge about how to select and apply operators, a working version of the Eight Puzzle has been demonstrated that maintains a continuous mental image of the board (Rosenbloom 2011c).

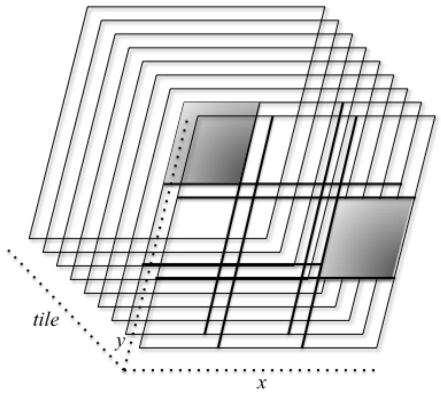


Figure 5: Partial visualization of the hybrid Eight Puzzle representation, with two continuous spatial dimensions (x and y) and one discrete dimension for tiles (tile 0 is the blank). The x,y region in which tiles sit are set to 1 (grey), while all others are 0 (clear). Only the blank and tile 1 are shown.

The Eight Puzzle is limited to square objects, which fit neatly within rectangular regions, but it is also possible to approximate more complex 2D shapes via multiple such regions. The quality of the approximation depends on the size of the regions into which it is decomposed, and thus also on the number of regions, inducing a tradeoff between the accuracy of the shape representation and its cost.

Just as the representation of a single object can involve multiple regions within an imagery function, the representation of different attributes of an object may require multiple imagery functions. For example, shape and color in a 2D image could each be represented as a distinct 3D Boolean function, rather than as a single 4D function. Such a representation encodes each attribute in a separate channel, as is found in primate vision (Livingstone and Hubel 1987), implicitly embodying the assumption that shape and color are conditionally independent given the location, and greatly reducing the combinatorics of the resulting representation. Yet, whether or not imagery is decomposed in this fashion, its representation is just a special case of the generalized, piecewise continuous representation used for WM, implying that distinct WMs should not be required for symbolic and spatial information.

Probability Distributions

Probability distributions require a step beyond the hybrid representation used for WM and mental imagery. Instead of Boolean functions, their functional values must range continuously over the interval [0,1]. A prior probability distribution over a single random variable requires only one dimension. The domain of the random variable is represented along the dimension, with the functional values at any point specifying its probability. The same approach also works directly for probability density functions over continuous variables; the single dimension is simply continuous rather than discrete and the function specifies densities rather than probabilities (Figure 6). For a conditional probability distribution, two dimensions are used, one for each random variable, with the functional value now specifying the conditional probability of one given the other.

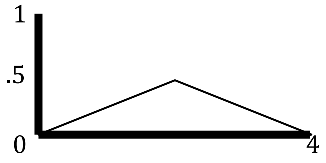


Figure 6: Two-region pyramidal probability density function.

Symbolic WM elements can be represented exactly by piecewise linear functions. Mental imagery, in contrast, requires approximations because of the orthotopic restriction on region boundaries. Probability functions in their turn require approximations because of the linear restriction on region functions. A Gaussian, or normal, distribution could be approximated as a constant function with one region, as a pyramid with two regions (Figure 6), or as accurately as desired with enough sufficiently small regions. This approximation thus also involves an implicit tradeoff between accuracy and cost.

With probability distributions added to general symbolic processing we see a bridging of the logical vs. probabilistic and data-driven vs. model-based dichotomies. These bridges are key to the implementation of semantic memory within the graphical architecture, and to work in progress that uses this memory in natural language tasks. They are also crucial to work in progress on bridging the central vs. peripheral dichotomy. Both of these variants of work in progress will be discussed in a bit more detail in the section on graphical models.

Episodic Memory

Episodic memory (EM) can be viewed as a sequence of temporally indexed instances of WM (Nuxoll and Laird 2007). In the graphical architecture there is one EM function per predicate. These EM functions are just like WM functions, but with an additional dimension for time. Regions in EM thus have a temporal extent, spanning any length of time over which the same linear function applies. Figure 7 shows the EM function learned for the selected operator in a typical run of the Eight Puzzle after 5 decisions, each of which defines one time step.

	1	2	3	4	5
Left	1		0		0
Right	0		0		0
Up					
Down	0		1		0

Figure 7: Episodic memory for the selected operator after 5 time steps. Left was selected first, Down for the next three steps, and then nothing (with extrapolation into the future).

In its simplest form, regionality implies that if the WM for a particular predicate doesn't change, then no regions need to be added to its EM. In fact, because the last temporal region extends to infinity, the EM need not be modified at all under such circumstances. A temporal boundary is only added when the existing final region is inadequate for the new content, automatically yielding the kind of compact representation that requires special purpose optimizations in other systems (Derbinsky and Laird 2010). It also directly enables a simple form of extrapolation, where the future is automatically assumed to be like the present.

Beyond the EM functions themselves, an additional aspect of EM that is also represented in a piecewise linear manner in the graphical architecture is a temporal decay function, encoded as a prior distribution on the temporal variable that prefers more recent memories. Rather than memories actually decaying, this temporal prior combines with the matching of cues in WM to episodes in EM to determine which episode is retrieved. With a piecewise linear representation, it is possible to represent linear decay via a single region, or exponential decay via a piecewise approximation. For the Eight Puzzle run above, a normalized linear function was used whose slope is automatically (re)set each time step to $2/(t^2-1)$, where t is the time steps that have so far elapsed.

Potential Enhancements

The manner in which the graphical architecture addresses these four representational challenges illustrates how a subsumption-based generalization reduction that embodies an appropriate compromise between expressiveness and efficiency can bridge the representational aspects of many dichotomies. Work in progress is exploring potential enhancements to both how regions are structured and how functions are defined on them, which together should improve the expressiveness and efficiency of the representation while strengthening the bridges already constructed for the discrete vs. continuous, symbolic vs. subsymbolic and central vs. peripheral dichotomies.

For region structuring, we are beginning with the elimination of several restrictions on the existing orthotope representation. Eliminating the restriction that regions must be axially aligned, and instead allowing orthotopes at arbitrary orientations, should facilitate mental imagery by: reducing the number of regions required to represent objects whose boundaries don't align with spatial axes, enabling objects to be rotated in mental imagery without reslicing them afterwards, and supporting compact representation of the kinds of shifted delta functions that underlie object translation (Rosenbloom 2011b). It should also directly support more general forms of extrapolation in EM; enabling, for example, automatic trajectory projection for a vehicle with constant velocity from an open-ended final region that lies diagonally across space-time (Figure 8). Eliminating the restriction that region boundaries must be part of function-spanning slices should reduce unnecessary region fragmentation; for example, avoiding the full horizontal slices between the operators in Figure 7, to yield Figure 9.

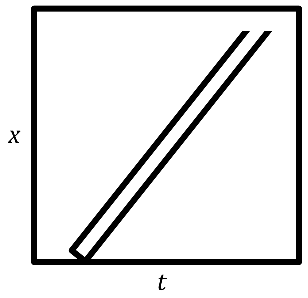


Figure 8: Notional 1D (x) location extrapolation for a vehicle with a constant velocity via an angled final space-time region in episodic memory.

	1	2	3	4	5
Left	1		0		0
Right	0				
Up					
Down			1		

Figure 9: What episodic memory from Figure 7 should look like when slices need no longer span the entire space.

Eliminating these two restrictions may, however, make processing more complex. With an array of axially aligned orthotopes, regions can be traversed systematically for computing sums, products, etc. Without these restrictions, region alignment becomes more like the computer graphics problem of collision detection for bounding boxes. It will become even more like the graphics problem with the elimination of the assumption that all regions are explicitly represented, enabling inactive regions – those with a value of 0 – to be implicit, much as inactive elements in normal WMs are implicit. This should yield a sparse function representation of active orthotopes at arbitrary orientations, greatly reducing the regions that need to be represented and processed.

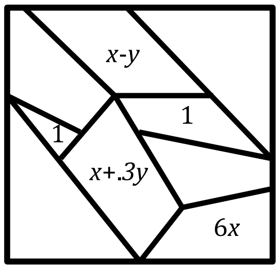


Figure 10: Piecewise linear function over 2D polytopes, with inactive regions implicit.

We are also exploring going beyond orthotopes, to *convex polytopes* – *N* dimensional generalizations of polygons (Figure 10). Once systematicity is lost, and region borders can run diagonally across dimensions, there is little efficiency gain in staying with orthotopes, and useful expressivity to be gained with polytopes. (Moreover, if active orthotopes can occur at arbitrary orientations, inactive segments may, at least implicitly, require decomposition into polytopes anyway.) Such a shift to polytopes will make the representation even more like that used in graphics, facilitating compact descriptions of complex objects, but also converting alignment into a more complex form of collision detection.

In moving beyond the expressivity of linear region functions, we are considering alternative parametric forms – such as polynomials and (mixtures of) Gaussians – along with nonparametric forms. These further broaden the continuous functions exactly representable while also yielding more compact and accurate approximations to other functions. However, they also further increase the complexity of the processing.

Graphical Models

Graphical models provide efficiency in computing with multivariate functions by decomposing them into products of simpler functions. Bayesian networks do this for joint probability distributions, yielding products of prior and conditional distributions. The graphical architecture uses factor graphs, which are similar to Bayesian networks, except that they: (1) decompose arbitrary multivariate functions; (2) utilize bidirectional links; and (3) include explicit factor nodes in addition to variable nodes to represent the subfunctions in the decomposition. In using bidirectional networks able to represent more than just probabilities, factor graphs are more like Markov networks (aka Markov random fields) than Bayesian networks, but they are more general than even the Markov alternative.

Factor graphs support more complex forms of representation than are provided directly by piecewise continuous functions. The piecewise functions define elementary representational elements, corresponding to instances of predicates and images, while factor graphs link them together in a manner that respects any independence assumptions among them. These more complex relationships are specified in the graphical architecture via *conditionals* that compile down to factor graphs (Rosenbloom 2010). Each conditional may contain *conditions*, *actions*, *condacts* and *functions*. Conditions and actions are as in standard rule systems; conditions match to WM to constrain which rules fire and to yield variable bindings, while actions modify WM. Condacts meld these functionalities to yield the bidirectional processing that is crucial in probabilistic graphical models. Functions in conditionals are specified over subsets of conditional variables, and compile down to functions in particular factor nodes within the graph.

This representation has been shown to be general enough to handle rules, facts, episodes, and constraints (Rosenbloom 2010). It also handles factored probabilistic representations based on products of prior and conditional probabilities, in support of naïve Bayes classifiers and other forms of probabilistic reasoning. The classifier that underlies the semantic memory (of facts) consists of a prior distribution on the object category plus conditional distributions on features given the category, enabling it to predict from arbitrary cued features an object's category and any uncued features. This memory has been leveraged directly for work in progress on question answering in virtual humans, as in [Leuski and Traum 2008], and word sense disambiguation. More general combinations of prior and conditional probabilities are also being used within the architecture to strengthen the existing bridge across the central vs. peripheral dichotomy, via work that combines perception, localization and decision-theoretic decision making (Chen et al. 2011).

Inference in factor graphs typically occurs by either sampling or message passing. The graphical architecture uses a variant of the summary product algorithm, which passes messages between variable and factor nodes about possible values of the variables. Incoming messages are (pointwise) multiplied together at all nodes, with factor nodes also including their functions in the product and then summarizing out all variables not in an output message to a variable node. When sum/integration is used for summarization, the graph computes marginals on its individual variables. When max is used instead, the graph computes the MAP estimation over all of the variables.

In the graphical architecture the summary product algorithm – with integration used for summarization in place of sum – is applied to the full representational generality provided by factor graphs and piecewise linear functions (with the latter specifying factor functions and messages). The result is a general implementation level that exemplifies the third form of reduction, providing a generalization below the architecture able to implement both sides of such dichotomies as procedural vs. declarative, logical vs. probabilistic, discrete vs. continuous, central vs. peripheral and hard vs. soft constraints. The use of factor graphs and the summary product algorithm at the implementation level, with the diversity of capabilities they enable in the architecture level above it, also exemplifies the second approach to reduction – where one side implements the other – in bridging the uniform vs. diverse dichotomy.

Conclusion

The articulation of three general desiderata for cognitive architectures that are to support virtual humans (and human-level cognitive systems) has led to a push to bridge a range of architectural dichotomies they induce. A strategy based on a pair of general architectural tools – graphical models and piecewise continuous functions – that leverage reduction for bridge building has been shown effective in bridging many of these dichotomies. In the process a novel path is opened up towards architectures that take a broad spectrum, tightly integrated, and functionally elegant approach to human-level intelligence.

Acknowledgements

This work has been sponsored by the U.S. Army Research, Development, and Engineering Command and the Air Force Office of Scientific Research, Asian Office of Aerospace Research and Development. Statements and opinions expressed do not necessarily reflect the position or the policy of the United States Government, and no official endorsement should be inferred. Thanks are due to the following collaborators: Abram Demski on new function representations; Sanjay Raveendran and Anton Leuski on question answering; Valay Shah and Abram Demski on word sense disambiguation; Junda Chen and Louis-Philippe Morency on perception; Teawon Han on localization; and Nicole Rafidi and David Pynadath on decision-theoretic decision making.

References

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., and Qin, Y. 2004. An integrated theory of the mind. *Psychological Review* 111: 1036-1060.

Bonasso, R. P., Firby, R. J., Gat, E., Kortenkamp, D., Miller, D. P., and Slack, M. G. 1997. Experiences with an architecture for intelligent, reactive agents. *Journal of Experimental & Theoretical Artificial Intelligence* 9: 237 – 256.

Chen, J., Demski, A., Han, T., Morency, L-P., Pynadath, P., Rafidi, N. & Rosenbloom, P. S. (2011). Fusing symbolic and decision-theoretic problem solving + perception in a graphical cognitive architecture. Submitted to the *2nd International Conference on Biologically Inspired Cognitive Architectures*.

Derbinsky, N. and Laird, J. E. 2009. Efficiently implementing episodic memory. In *Proceedings of the 8th International Conference on Case-Based Reasoning*.

Domingos, P. and Lowd, D. 2009. *Markov Logic: An Interface Layer for Artificial Intelligence*. Morgan & Claypool.

Kieras, D. E. and Meyer, D. E. 1997. An overview of the EPIC architecture for cognition and performance with application to human-computer interaction. *Human-Computer Interaction* 12: 391-438.

Koller, D. and Friedman, N. 2009. *Probabilistic Graphical Models: Principles and Techniques*. Cambridge, MA: MIT Press.

Kschischang, F. R., Frey, B. J. and Loeliger, H.-A. 2001. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47: 498-519.

Laird, J. E. Extending the Soar cognitive architecture. 2008. In *Artificial General Intelligence 2008: Proceedings of the First AGI Conference*. Amsterdam, Netherlands: IOS Press.

Laird, J. E., Newell A. and Rosenbloom, P. S. 1987. Soar: An architecture for general intelligence. *Artificial Intelligence* 33: 1-64.

Laird, J. E. and Rosenbloom, P. S. 1990. Integrating execution, planning, and learning in Soar for external environments. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, 1022-1029. Cambridge, MA: MIT Press.

Leuski, A., and D. Traum. 2008. A statistical approach for text processing in virtual humans. In *Proceedings of the 26th Army Science Conference*.

Livingstone. M. S. and Hubel, D. H. 1987. Psychophysical evidence for separate channels for the perception of form, color, movement, and depth. *The Journal of Neuroscience* 7: 3416-3468.

Nuxoll, A. M. and Laird, J. E. 2007. Extending cognitive architecture with episodic memory. In *Proceedings of the 21st National Conference on Artificial Intelligence*.

Rosenbloom, P. S. 2010. Combining procedural and declarative knowledge in a graphical architecture. In *Proceedings of the 10th International Conference on Cognitive Modeling*.

Rosenbloom, P. S. 2011a. Rethinking cognitive architecture via graphical models. *Cognitive Systems Research* 12: 198-209.

Rosenbloom, P. S. 2011b. Mental imagery in a graphical cognitive architecture. In *Proceedings of the 2nd International Conference on Biologically Inspired Cognitive Architectures*. In press.

Rosenbloom, P. S. 2011c. From memory to problem solving: Mechanism reuse in a graphical cognitive architecture. In *Proceedings of the 4th Conference on Artificial General Intelligence*. In press.

Rosenbloom, P. S., Laird, J. E. and Newell, A. 1987. Knowledge level learning in Soar. In *Proceedings of Sixth National Conference on Artificial Intelligence*, 499-504. Menlo Park, CA: AAAI.

Sun, R. 2006. The CLARION cognitive architecture: Extending cognitive modeling to social simulation. In R. Sun (Ed.), *Cognition and Multi-Agent Interaction*. New York, NY: Cambridge University Press.