

Self Organizing Feature Maps for Logical Unification

A. Ultsch, G. Guimaraes, V. Weber
University of Marburg,
Department of Mathematics,
Hans-Meerwein-Straße
35032 Marburg
Germany

Abstract

Unification plays a central role in logic programming (e.g. Prolog) and is also a central feature for the implementation of modern expert systems. In this work we present an implementation of the unification algorithm using Artificial Neural Networks. This results in a parallel and efficient implementation even for important tests like occur and clash. Our approach is such that besides the exact unification as needed in logic programming an approximative (fuzzy) unification with certain properties becomes possible. Inexact unification may be used for error tolerance as well as analogical and approximative reasoning in expert systems. The main idea of our approach is to use Kohonen Feature Maps (KFM) for the representation of terms. In KFM similar input data is represented in a close neighborhood in the feature map. A special encoding therefore maps contextual semantic relationships to topological neighborhoods. Keywords: Automated Reasoning, Unification, Fuzzy, ANN, Unsupervised Learning

1. Introduction

Unification plays a central role in logic programming (e.g. Prolog) and is also a central feature for the implementation of modern expert systems. In this work we present an implementation of the unification algorithm using Artificial Neural Networks (ANN).

Unification is the process of making two terms equal. Consider, for example, the following equation in the variables X and Y:

$$f(g(a,X))= f(g(Y,Y))$$

This equation can be solved if the atom a is identified with Y and X with Y. A successful unification yields therefore:

$$f(g(a,a))= f(g(a,a)), Y=a, X=Y, X=a.$$

The aim of an unification algorithm is to find the most general unificator (mgu). This is to find the most general instance for the variables in the terms, such that the two terms become equal. There are two important reasons that two terms may not be unifiable. First, it may be necessary that in order to make the terms equal two different atoms or functors have to be identified. This is logically not possible and called a Clash. Second a term may be subterm of itself like in the equation $X = f(X)$. This would result in an infinite unification process $X = f(f(f(f(f(f(\dots$. A test for this is called Occurrence-check. Almost all implementations of unification include a Clash test. Many unification algorithms, however, omit the occurrence test, for pragmatic reasons [STERLING/SHAPIRO 88] since this test is costly and slows the

unification algorithms down significantly. The network presented in the following chapters includes the occur check during the unification process without time delay.

Our interest is to accelerate the unification process such that correct substitutions are estimated but not searched among all possibilities. Using Artificial Neural Networks (ANN) for unification this can be achieved. Several systems for unification using ANN have already been proposed [AJJANAGADDE/SHASTRI 89, BALLARD 86, HÖLLDOBLER 90, STOLCKE 89, TOURETZKI/HINTON 89]. In our approach we use KFM for the representation of the atoms/functors to enable similarity between components. Actual unification is performed in a relaxation network that is architecturally close to that proposed by Hölldobler [HÖLLDOBLER 90].

2. Kohonen Feature Maps for the Representation of Terms

The main idea of our approach is to use Kohonen Feature Maps (KFM) [KOHONEN 84] for the representation of atoms and functors in a term. KFM have the property that similar input data is represented in a close neighborhood in the feature map. This may be seen, for example, in the work of Ritter and Kohonen shown in Figure 1 [RITTER/KOHONEN 89]. Entities like birds, cattle and hunters are represented on neighboring regions of the KFM. Subclasses like the dog-like animals "wolf dog fox" and cat-like animals "tiger lion cat" can also be found. In this experiment 16 different animals were used as input for a 10x10 KFM. Figure 1 shows the best matches for these input vectors.

⊠

Fig. 1: Representation of animals in a KFM in [RITTER/KOHONEN 89]

The aim to use a KFM for the representation of terms is to encode the elementary parts of terms, i.e. atoms and functors such that their semantical context is mapped onto a topological neighborhood on the feature map. Ritter and Kohonen describe an algorithm to get n-dimensional inputs clustered on a "semantic" map, using words of sentences in a grammatical and semantical context. Their main idea is to represent (for example) the symbols and the attributes in a due context and then weight the two parts properly such that the attributes predominate in the learning process [RITTER/KOHONEN 89]. The due context could now be seen as a vector comprising a "symbol field" and an "average word context". The context of a word is then formed both by the predecessor and the successor, or only by the immediate predecessor. In such a way words would only be represented in the semantic context in which they are used. After the codification the authors use the concatenation of both, symbol vector and attribute vector as input vector for the feature map.

In our approach the atoms and functors from the literals of the knowledge base are matched on a KFM during a learning phase. Here for the codification of each atom resp. functor a vector (as input for the Input Feature Map) is generated as follows: the length of the vector is the number of possible atoms and functors [WEBER 92]. Each component of each input vector represents the number of occurrences of the given atom/functor in a (sub-) term. Figure 2 shows the codification of a possible Prolog knowledge base. For each component, here we have 3 functors and 5 atoms, a vector is generated such that the occurrences of the other components in the terms of the database are set to the number of occurrences. The occurrence of the same component and all the rest of the components are set to "0". For example, for the functor f the following components are considered: g, h, a, b, c . So we get the vector $f=(0,1,1,1,1,0,0)$. The length of the vectors corresponds to the number of components in the database. In our case we have eight.

To describe the components we use the following notation: the number after the component refers to the arity of the component, for example the functor $g/2$ has the arity "2". Now we want to make a reference to another important concept used for the codification of the components in the network, the argument position. Each level of the main term gets a new number: 0,1,2,... The terms on the argument positions of each level are again subdivided. In our example $f(g(a,X))$, we get the following argument positions: !! EINBETTEN "Equation"

Ð

Fig.3.: Unification Network

Units in the Variable Vector are connected to a Pairs unit with an interneuron implementing a logical AND function. This means that a Pairs unit is activated only if it is required that variables at the corresponding argument positions have to be unified. The activation of such a Pairs unit is transmitted to the corresponding Cube layer and back to the Variable Vectors via units implementing a logical OR function.

As an example consider the unification of $f(g(a,X)) = f(g(Y,Y))$. First the components of the terms are encoded as mentioned before. As argument positions we have in the term: $f/0$, $g/1$ and $a/1.1$. We also need two Variable Vectors for the variables X , Y . Figure 3 shows how the Variable Vectors, Pairs and Cube are connected.

The activation functions of the different units are constructed such that if the network has reached a stable state (relaxation process), the unification process can be performed. The activity of the Pairs after a propagation indicates that the stability of the network is not reached. In this case, the two units of the Variable Vectors are active or an activation between the Pairs exists. This unification network was designed and implemented by Volker Weber. For details see [WEBER 92].

In order to actually calculate the mgu a KFM, called Output Feature Map, is constructed. Weights of this feature map are the activations of the Cube units. If the Variable Vector of a variable is used as input pattern to the Output Feature Map, the unit representing an instance of that variable responds. In the case of our example using the Variable Vector for Y as input to the Output Feature Map the unit located at the position of a is activated. This means $Y = a$. In the same way $X = Y$ and $X = a$ can be derived.

In our network tests like occurrence and clash are implemented such that they can be calculated in parallel and during the relaxation process. Unification is performed via a relaxation network. If this network has reached a stable state the mgu can be read out using a KFM. It can be proven that our network performs the unification process precisely. Besides that the usage of KFM gives the opportunity to do approximative reasoning as described in the next chapter.

4. Fuzzy Unification

It can be proven that the network construction described in the previous chapters performs exactly the unification process needed for logic programming [WEBER 92]. This means, that the proposed network is an ANN realisation of the unification algorithm. The merit of this being is that a fast and parallel implementation is possible in particular on a parallel hardware like transputers, connection machines, etc.[GUIMARAES/KORUS 92].

Real world applications of logic programming, in particular in expert systems, require more than exact reasoning capabilities. Many empirical predicates can not be characterised as being true or false but rather as being more or less probable or possible. If human users construct knowledge bases, different words could be used for the same term like "neighbor" and "neighbour". Our system would detect the similarity of the two terms by their topological closeness in the KFMs.

In order to perform fuzzy unification the AND and OR units of the relaxation networks have to be modified. Instead of the AND function in the units with connections from the Variable Vectors to the Pairs units the activation function is changed to the minimum of the two input activations. In the case of the units with input from the Pairs units and outputs to the Variable Vectors respectively the layers of the Cube instead of the OR- function the maximum of the input activations is propagated.

5. Applications

We tested our System with the following program types:

- 1.) A small database system written in Prolog with about 5000 entries. The entries are literary references.
- 2.) Programs for the simplification of algebraic terms and for symbolic differentiation. The programs consists of about 50 typical Prolog facts and rules with functors having no more than 4 arguments.
- 3) A program to solve the Traveling Saleman Problem for some german cities.

A detailed description of the results can be found in [WEBER 92]. A summary of the performance of our algorithm applied to all programs is shown in the next figure. The mean relative time needed to perform all the necessary unifications is shown.

Reference standard is Prolog's unification algorithm requiring in general an effort that is quadratic in the length of the terms. Infinite unification, as proposed in [HUET 76] requires in many cases a linear effort, in some cases, however this approach is also quadratic. The fastest, compared to our approach 0,005% slower, is an algorithm for linear unification as published in [DWORK ET AL 84].

Figure 4: Mean relative running time for different unification algorithms

6. Conclusion

In this work a connectionist unification algorithm was presented. For the encoding and decoding of terms KFM are used. The system is able to do unification in parallel and can be proven to be an exact realisation of an unification algorithm.

The usage of KFM for the representation of terms allows, however, to do also inexact (uncertain, fuzzy) reasoning. In KFM similar input data are represented in a close neighborhood in the feature map. A special encoding therefore maps contextual semantic relationships to topological neighborhoods.

Therefore our system allows to generalise the unification algorithm to a more flexible unification strategy where syntactic closeness, such as in the presence of typographic errors, and semantic closeness (same usage of terms, analogical reasoning) are respected. This type of unification could play a central role in systems where an algorithm for approximative reasoning, like in expert systems, is needed. Experiments with several examples stemming from quite different application areas such as symbolic differentiation or a Prolog program for the travelling salesman problem show the practicability of our approach.

We are aware, however, that for a truly connectionist realisation of unification the one-to-one correspondence of terms to units in the network (local representation) should be abandoned in favour of a distributed representation. We will investigate such an approach in the near future. The main problem here is an appropriate encoding of the concatenation and aggregation of terms such that a decoding is possible. Preliminary results in this area are, however, promising [WEBER 92].

References

- [AJJANAGADGE/SHASTRI 89] Ajjanagadde, V., Shastri, L.: Efficient inference with multiple predicates and variables in a connectionist systems, Proc. Annual Conf. Cognitive Science Soc., 1989, pp 396-403.
- [BALLARD 86] Ballard, D.H.: Parallel logic inference and energy minimization. Proceedings of the AAAI Natl. Conf. on AI, Philadelphia PA, pp 203-208, 1986.
- [DWORK ET AL 84] Dwork, C., Kanellakis, C., Mitchell, J.C.: On the sequential nature of

unification. In: Journal of Logic Programming, 1, pp.35-50, 1984.

[ESCALADA-IMAZ/GHALLAB 88] Escalada-Imaz, G., Ghallab, M.: A practically efficient and almost linear unification algorithm. In: Artificial Intelligence, 36(2), pp.249-263, 1998.

[GUIMARAES/KORUS 92] Guimaraes, G., Korus, D.: Neuronale Netze auf Transputern. In: Abstraktband TAT«92 Klinikum der RWTH Aachen, September 1992.

[HÖLLDOBLER 90] Hölldobler, S.: On high level inferencing and the variable binding problem in connectionist networks, in Dörfner G. (ed.): Konnektionismus in Artificial Intelligence und Kognitionforschung, Springer, pp 180-185, 1990.

[HUET 76] Huet, G.: Resolution d`equations dans les langages d`ordre 1,2,..., !!

EINBETTEN "Equation" * mergeformat ¶ Ð½. PhD thesis, Universté de Paris VII, 1976.

[KOHONEN 84] Kohonen, T.: Self-Organization and Associative Memory. Springer Series in Information Science. Springer Verlag, Berlin, 1984.

[LI/LIU 90] Li, D., Liu, D.: A Fuzzy Prolog Database System. Computing System Series. Research Studies Press Ltd., Taunton, Somerset, England, February 1990.

[PATERSON/WEGMAN 78] Paterson, M.S., Wegman, M.N.: Linear Unification. In: Proceedings of the Symposium on the Theory of Computing, 1976.

[RITTER/KOHONEN 89] Ritter, H., Kohonen, T.: Self-Organizing Semantic Maps, in: Biological Cybernetics, 61, pp.241-254,1989.

[RITTER 90] Ritter, H.: Self-organizing maps for internal representations. in: Psychological Research, 52, pp.128-136,1990.

[STERLING/SHAPIRO] Sterling, L., Shapiro, E.: The Art of Prolog. MIT Press, Massachusetts, 1988.

[STOLCKE 89] Stolcke, A.: Unification as constraint satisfaction in structured connectionist networks, Neural Computation, 4, 1989.

[TOURETZKI/HINTON 89] Touretzki, D.S., Hinton, G.E.: A distributed connectionist production system, Cognitive Science 12, pp. 423-466, 1989.

[ULTSCH ET AL 90a] Ultsch, A., Hannuschka, R., Hartmann, U., Mandischer, M., Weber, V.: Connectionist represented control knowledge. In Proceedings of the 3rd Neuro-Nimes, pp.559-561, November 1990.

[ULTSCH ET AL 90b] Ultsch, A., Hannuschka, R., Hartmann, U., Weber, V.: Learning of control knowledge for symbolic proofs with backpropagation networks. In Eckmiller, R., Hartmann, G., Hauske, G., editors, Parallel Processing in Neural Systems and computers, pp.499-502, Amsterdam, North-Holland, March 1990.

[ULTSCH ET AL 90a] Ultsch, A., Hannuschka, R., Hartmann, U., Mandischer, M., Weber, V.: Optimizing symbolic proofs with connectionist models. In Kohonen, T., Mäkisara, K., Simula, O., Kangas, J., editors, Artificial Neural Networks, volume 1, pp.585-590, Amsterdam, North-Holland, June 1991.

[ULTSCH ET AL 90a] Ultsch, A., Hannuschka, R., Hartmann, U., Mandischer, M., Weber, V.: Control of PROLOG-Proofs using connectionist models. In Bazewicza, editors, Information Systems Architecture, ISAT«91, pp. 175-183, Wroclaw, Politechnika Wroclawska,1991.

[ULTSCH 91a] Ultsch, A.: Konnektionistische Modelle und ihre Integration mit wissensbasierten Systemen, Internal Report Nr. 396, University of Dortmund, Germany, Februar 1991.

[ULTSCH 91b] Ultsch, A.: Eine Einordnung von Kohonen Netzen in eine Systematik für Konnektionistische Modelle und ein Beitrag zu ihrer Analyse, Internal Report Nr. 406, University of Dortmund, Germany, Dezember 1991.

[WEBER 92] Weber, V.: Unifikation in Prolog mit konnektionistischen Modellen. Diplomarbeit, University of Dortmund, Germany, Informatik, 1992.

ëÄð;Ðôðú5[↓] ù5[↓] æh.¥'A¶àßà'¶©¶- |ý[↓] HH[↓]
ð²ÿâÿä[↓] +ð6[↓] G | {[↓] fiðHH[↓]
ð²ÐdÐÐÐÐðÐ¶ ÐÐð³Ðê@Ð ◀ :ÿä+ð¼⁻¾\oe þËË<<Ð[↓] S¶ ¶ &- ¶ -ÿÿÿ[↓] ¶ Word

Microsoft Word

-
Đ0ú

Êð -

Ð0úÝÊðx-

Ð0úÝÊðÆ- ☿ Ð0úÐÄËÐà- ☿ Ð0úÐÐÄË^{ℓ ℓ} Ä- ☿ Ð0úÐÐÀËÝ- ◀ Ð0ùàÎ^ℓ -- ◀ Ð0ù^ℓ pðÎ<
- ←^ℓ 0ÐÄ† ü | 8^ℓ ³YðÐÌÖùø- -^ℓ 0Ä† ü<ð¥çÄÆðÐRTüÐà-^ℓ 0Ð'Öü -pð¥!! Ä8^ℓ ÀpÐ† T
- ÝÆ-)'0ðiT

pđŸ 8^l ÀpĐR(- Ÿ - † 0đi^{3/4}

ÝpÝÙ1ÄöĐâ(ü>- !! ʘ 0điTüĐÄÍʘ x- !! ʘ 0ĐÊñüĐĐÀÍà- † Đ0øâÍ| ʘ À- † Đ0øpÍŕ ÆĐ- †
Đ0ø8Í<ÆĐ- † Đ0ø ÍøÆĐ- ◀ Đ0øŕ İĐàÆĐ- ◀ Đ0øĐÝÄİ-
ÆĐ- ◀ Đ0øĐÄİ<ÆĐ- ◀ Đ0øĐĐàÌÆÆĐ- ◀ Đ0ÆxÓĐÝÀÆĐ- ¶ Đ0Æ<ü`ÙüĐ½Đ0Æü
öĐÝüüĐ¾Đ0ÆĐŕ ÄÆđr2 öĐ?äüĐ³Đ0ÆĐÝàÆđ½+PÆđĐýÄüĐ²Đ0ÆĐĐÆÆđ*0ÆĐ-
øÆ ʘ p-)Ý0à

Æ

p|^L ÄD*PÐÀ ÆáyÄÝÆ^L p-)Ý0à

Æ

p>ÄsrhDÀ ÆÿàÝÆ^l p- ¼^l 0âûðpç ÛDDÿùD- ↑ D0öDÝÄÛD?øùD- ↑ D0öDDàöDç ÆøD
- ↑ D0öÆöðDÿàøD- ◀ D0ö|öD?øÆD- ◀ D0ö>ÆD-
ÿöD- ↑ D0öç øðDÿàöD- ↑ D0öDÝÄùDøöD- ↑ D0öDDàúDÝÿÛD- ◀ D0ÛxúDÿÄÛD- ◀ D0
Û<úDÝüÛD- ◀ D0Û-úD?ÄÛD- ◀ D0Ûç üDDÆÚD½D0ÛDÝÄÆDç Àöý- ¶ D0ÛDàÆ?Û
Ä- ¶ D0ÛøÆDDøÛÛÈ@- ¶ D0Û<ÆDÝÄÛäUÄ D0öPÄ-8ÄÄ Æ àÝä]-)Ý0àÝÆ

p^L Äÿ 8~ĐÀ Æ àÝâë- -Ý0àÝÆ p^L ÄÝÄĐøÛ[↓] _ç- !! Đ0ÛĐÀÝÀÛđ@- !! Đ0ÛĐÆ-
ÄÛÄ-

ΔΟΥΔÆ>ÔÐ-

ΔΟΥΔ8φÔΔ-

Đ0Úđ-ĐàÔĐ-

ΔΟΥΔ

ÝÀÔ

Đ0ÚĐÝüÓ

Đ0ÚĐüÓ

Đ0ÚĐĐøÓĐĐ00àÓĐĐ00pÓĐĐ008ÓĐĐ00<ÓĐĐ00-ÓĐĐ00

ÓÐþÐ0ÒÝÓÐ¼ | 0ö1êðÄüÐÐðöÐ½ | 0Ð)J(ðÐÐÀû öÐ- -
| 0Ð)I8ðÐÐÀÆy²Ø°àÝÆ^l 0-)Ý0...H ÝÆ½p^l Ä¾àÐÀ
æ àÝÆ^l 0- &Ý0ÐÆ¾¾ÝÆ↑ p^l Ä¾pÐÀ æ àû^l 0- ¶^l 0Ð
ÚxÆð | \$ öÐ- ¶^l 0ÀÚ8Æð³ÆöÐþÐ0Æ¾¾ÔÐþÐ0Æ-ÔÐþÐ0Æ

ÔÐþÐ0ÆÝÔÐþÐ0ÆÝÔÐ

Ð0ÆÐÄÆÐ

Ð0ÆÐÄÆÐ

Ð0ÆÐÐÀÆÐ

Ð0ÆÐÐÀÆÐþÐ0ÔàÆÐþÐ0ÔÆÆÐþÐ0ÔpÆÐþÐ0ÔpÆÐ- † Ð0öð- † Æ8ÆÐ- † Ð0öðð
Æ8ÆÐ- !! Ð0ö^{L L} !! JÄÆ ÆÐ Ð0öðð©UÆ| ÐÀ

ÆàÝÆ^l 8-)Ý0`ÝÆ

$p \delta V < p \Delta A$

ÆàÝÆ^l 8- !Ý0`ÝÆppð·¥<

ÐÀààÐ- !! Ð0öÝÙ×ÄÆ

ÆÐþÐ0ÔÝÆÐþÐ0ÔÝÆÐþÐ0ÔÝÆÐ

Ð0ÔÐÄÒÐ

Ð0ÔÐÄÒÐ

Ð0ÔÐÄÒÐ

Ð0ÔÐÄÒÐ

Ð0ÔÐÄÒÐ

Ð0ÔÐÄÒÐ

Ð0ÔÐÄÒÐþÐ0ÓÀÒÐ- ¶ Ð0ÓàÙ[↓] 2- ¶ Ð0øÐÐÄùàÙ[↓] † - !! Ð0ÆÄùàÙ^{¼g-} ½Ð0ùðþÖÄ
ùàÙ[↓] † íÄ- ← Ð0ùð† TÄùàú àÝ† íÄ-)^{30`}Ý† TÄÄ àÀ

Æ àÝ† íÄ-)³0`Ý† (ÄÄ pÀ

ÆÆÝÆ[⊥] 8-)Ý0`ÝÆ

8^L Ä ^L ÄÄ

ÆÐþÐ00

ÆÐþÐ00

ÆÐþÐ00

ÆÐþÐ0Ô ÆÐþÐ0Ô ÆÐþÐ0Ô ÆÐ- † Ð0ÛþÆ8ÆÐ- † Ð0ÛÆ8ÆÐ- † Ð0öðàÛÄÆ8Æ
Ð³Ð0öÐU% @ÆpöÝÆ^l 8-)Ý0`ÝÆ

8ĐÍ%À pà

ÆàÝÆ^l 8- \$Ý0`ÝÆ

8ᐃᑭ ᐱ ᐱ

ÆàùÐ- † Ð0øδÛ\OE ÀÆàÆÐ- ◀ Ð0øÐÐ ÆàÆÐ- † Ð0ÛÀÆàÆÐ
Ð0ÆÐÐÀÆÐ
Ð0ÆÐÐÀÆÐ
Ð0ÆÐÐÀÆÐ
Ð0ÆÐÄÆÐ
Ð0ÆÐÄÆÐ
Ð0ÆÐÝÄÆÐþÐ0ÆÝÔÐþÐ0ÆÝÔÐþÐ0Æ

ÔÐþÐ0Æ

ÔÐþÐ0Æ

ÔÐ- † ˆ 0`Ú ÆÙÐ- ¶ ˆ 0 Ú ÆýöÐˆ 018Û<ÆðþÍÄøˆ ...Äˆ 0*î8ÆÐ
%øÐU- ˆ 0+îÛxÆÐ "Æ ÆÝÐ
-)Ý0*¶ ÝÆ

8^L Ä pà %Æ ÆÝÐU- 'Ý0}æÝÆÛ 8^L Ä pàpÍÄüÝÆöÄpÐ0ÆÆÔÐpÐ0ÆàÔÐpÐ0ÆÀÔÐ
Ð0ÒÐÄÀÔÐ
Ð0ÒÐÄÀÔÐ
Ð0ÒÐÄÀÔÐ
Ð0ÒÐÝÄÔÐpÐ0ÒÝÓÐýÐ?ÆýÐÆýÐ?ÆýÐÆðÙðÙðÙðÙðÙðÙ.1.%3:ýùBPAø%à\$ÈÈ
Ð^L S¶ ¶ &- Û -ÿÿÿ^l ¶ Word

Microsoft Word

&- ð ↑ Wordýýý Đá1 | Þðþð Ý← ↓ | | ¶ ð | !! ð | | ¶ ð | | !! ð 32 @@\$\$Sie benötigen
Word 6.0c oder höher, 32 ÄĐ@Sum Macintosh-Grafiken anzuzeigen.

↓ 1áýýýýĐÆĐÄÄ◄ Đ Ç å åĐ

ýýýýĐÆĐÄ1ĐÄÆĐñæÝðð0ĐÄÆĐó;Ý"Đ°¶ ;ððð üq²Đñ¶Đ°ÆĐñ¶Đ°¶ĐñÆĐñ¶Đ

Ýðð"Đñ¶| ²##+ Ê##à#Ñ £Đ ýýýýĐÆĐÄÑ "Điÿ+ ²#+ Ê#à ; ç 1.Đ← /ĐÝĐĐ

ýýýýýýýý8 å1BåRùÝðð0AäsúÝ"J´ ;ððð üq²:úZ´:úJ´Zú:úĐ Ýðð":ú¼+ ##Í+ ##à#Ñ £Đ

ýýýýĐÆĐÄÑ "9ù¼+ #Í+ #à ; çÝ"L^ ;ððð ü "UªUªUªUq-LRå^[RL^]^åR[RD ÝĐĐ

ýýýýýýýý"[RpÒ##1##Ûð ##\oe #ªUªUªUªUÑ £Đ ýýýýĐÆĐÄÑ #þÒ#1#Ûð #\oe

;ñþðð;öý^L (Đ - -S,ý¶ | Times^L ¶ ↓ Đ

þ.[↓] Đ+Đð ÝInput

*þÞFeature Map ó åĐ ýýýýĐÆĐÄÝ"1R ;ðð

Đ ÝĐĐ"Đ}0##[##+Æ##- # £Đ ýýýýĐÆĐÄ#0#[##+Æ#- ;Ý"Dz ;ðð Đ

ÝĐĐ"Đ}Æ##0##[##^L # £Đ ýýýýĐÆĐÄ"Đ}Æ#0#[#^L ;Ý"{} ;ðð Đ ÝĐĐ"[R+Í# £Đ

ýýýýĐÆĐÄ"[R+Í ;Ý"!! } ;ðð Đ ÝĐĐ"CR+−# £Đ ýýýýĐÆĐÄ"CR+− ;Ý"=} ;ðð Đ

ÝĐĐ"tR+...# £Đ ýýýýĐÆĐÄ"tR+... ;Ý"jk ;ðð Đ ÝĐĐ"¼kT# £Đ

ýýýýĐÆĐÄ"¼kT ;Ý"ˆt ;ðð Đ ÝĐĐ" tT# £Đ ýýýýĐÆĐÄ" tT ;Ý"}^ ;ðð Đ ÝĐĐ""^[# £Đ

ýýýýĐÆĐÄ""^[;çÝ"4^ ;ððð^L "ªUªUªUªUq-¼^4k% ^4^(k¼k% ^Đ ÝĐĐ ýýýýýýýý"% ^ð ##

Û##Ó##Ûð #Ñ £Đ ýýýýĐÆĐÄÑ #ð #

Û#Ó#Ûð ;ñþðð;öýýüýĐ

X³j[↓] (¼Y^L a/0 ó;ñþðð;öýýýĐ a9ÖL(j:↓ g/2

*þĐ

*þ^L f/1 óĐ ýýýýĐÆĐÄÝ"9\OE ;ððð ý ◄ "Dà◄ "Dàq&Ý\OE íĐLYÆ9\OE í\OE

íĐ#9Đ#ÝĐLYÆĐ ÝĐĐ ýýýýýýýý"ÝÆÆ2##Y##U##B##)\OE ##◄# ◄ "Dà◄ "DàÑ £Đ

ýýýýĐÆĐÄÑ #Æ2#Y#U##B##)\OE #◄ ;Ý"ÝĐý ;ððð ý q-ÝB8Đ← 8BÝĐýÝĐ← 8Ú8BĐ ÝĐĐ

ýýýýýýýý"8B)\oe ##!! ##Æ1##Í# Ñ £Đ ýýýýĐÆĐÄÑ #)\oe ##!! #Æ1#Í ; 18ÇiÛ ýýýýýýýý8

18åö ýýýýýýýý8 à"à"à"à"18úĐ

ÝÐÐ"°F+# £Ð ÿÿÿÐÆÐÂ"°F+ ;Ý"üq ;¸ð ÆÐ ÝÐÐ"üF+# £Ð
ÿÿÿÐÆÐÂ"üF+ ;Ý"Ð◀q ;¸ð ÆÐ ÝÐÐ"Ð◀F+# £Ð ÿÿÿÐÆÐÂ"Ð◀F+ ;;ñþÐð;öýÿüÝÐ
ÆRÆc(¶S^L 0,1 ó;ñþÐð;öýÿÆþÐ ÂO\OE j(ÀP| 0,1.1 ó;ñþÐð;öýÿüþÐ
ÆOãj*½| 0,1.2 ó;ñþÐð;öýÿÆþÐ ÎOøj*½| 1,1.1 ó;ñþÐð;öýÿüþÐ ÐÐOÐ
j*½| 1,1.2 ó çÐ ÿÿÿÐÆÐÂ QÐ¶ àÐ!i ÿÿÿÿÿÿÿX QÐ0£Ð< ÿÿÿÿÿÿÿX Qö£Ð<
ÿÿÿÿÿÿÿXY"Ð5Ðt ;¸ð üÐ ÝÐÐ Ð5- Ð5Ðt##!! # £Ð ÿÿÿÐÆÐÂ
Ð5- Ð5Ðt##! ;Ý"ÐHÐ[;¸ð üÐ ÝÐÐ"Ð5Ð[!! # £Ð ÿÿÿÐÆÐÂ"Ð5Ð[!! ;Ý"ÐHù ;¸ð üÐ
ÝÐÐ"Ð5ù!! # £Ð ÿÿÿÐÆÐÂ"Ð5ù!! ;Ý"ÐHà ;¸ð üÐ ÝÐÐ"Ð5à!! # £Ð
ÿÿÿÐÆÐÂ"Ð5à!! ;Ý"ÐOÐt ;¸ð üÐ ÝÐÐ"ÐFÐq^L ##^L Æ# £Ð ÿÿÿÐÆÐÂ"ÐFÐq^L
#^L Æ ;Ý"ÐOÐ[;¸ð üÐ ÝÐÐ"ÐFÐYð ##^L Æ# £Ð ÿÿÿÐÆÐÂ"ÐFÐYð #^L Æ ;Ý"ÐOù ;¸ð
üÐ ÝÐÐ"ÐFö ##^L Æ# £Ð ÿÿÿÐÆÐÂ"ÐFö #^L Æ ;Ý"ÐOà ;¸ð üÐ ÝÐÐ"ÐFÆ ##^L Æ# £Ð
ÿÿÿÐÆÐÂ"ÐFÆ #^L Æ ;Ý"Ð4- ;¸ð üÐ ÝÐÐ"Ð1,Æ^L ##^L # £Ð ÿÿÿÐÆÐÂ"Ð1,Æ^L #
;Ý"Ð2â ;¸ð üÐ ÝÐÐ"Ð2t½# £Ð ÿÿÿÐÆÐÂ"Ð2t½ ;Ý"Ð2p ;¸ð üÐ ÝÐÐ"Ð1xø^L ##ý^L # £Ð
ÿÿÿÐÆÐÂ"Ð1xø^L #ý ;Ý"Ð2Ðz ;¸ð üÐ ÝÐÐ Ð2iÐ2Ðz##4# £Ð ÿÿÿÐÆÐÂ
Ð2iÐ2Ðz#4 ;Ý"ÐNÐb ;¸ð üÐ ÝÐÐ"Ð2Ðb4# £Ð ÿÿÿÐÆÐÂ"Ð2Ðb4 ;Ý"ÐNÿ ;¸ð üÐ
ÝÐÐ"Ð2ÿ4# £Ð ÿÿÿÐÆÐÂ"Ð2ÿ4 ;Ý"ÐNÊ ;¸ð üÐ ÝÐÐ"Ð2Ê4# £Ð
ÿÿÿÐÆÐÂ"Ð2Ê4 ;Ý"Ð2i ;¸ð üÐ ÝÐÐ"Ð1ùø^L ##ý^L # £Ð ÿÿÿÐÆÐÂ"Ð1ùø^L #ý ;Ý"Ð
É ;¸ð üÐ ÝÐÐ"Ð q↑ ##½# £Ð ÿÿÿÐÆÐÂ"Ð q↑ #½ ;Ý"Ð¶ É ;¸ð üÐ ÝÐÐ"Ð¶ q↑ ##-
Î# £Ð ÿÿÿÐÆÐÂ"Ð¶ q↑ #Î ;Ý"Ð5£ ;¸ð üÐ ÝÐÐ"Ð.ùÝ##ÆÆ# £Ð
ÿÿÿÐÆÐÂ"Ð.ùÝ#ÆÆ ;Ý"Ð£ ;¸ð üÐ ÝÐÐ"ÐÝù##ÆÐ# £Ð
ÿÿÿÐÆÐÂ"ÐÝù##ÆÐ ;Ý"üÐ< ;¸ð üÐ ÝÐÐ"ü l##Æ# £Ð
ÿÿÿÐÆÐÂ"ü l#Æ ;Ý"°Ð ;¸ð üÐ ÝÐÐ"üÐÆ# £Ð ÿÿÿÐÆÐÂ"üÐÆ ;Ý"-Ð ;¸ð üÐ
ÝÐÐ" Ððø##^L ý# £Ð ÿÿÿÐÆÐÂ" Ððø#^L ý ;Ý"-Ð< ;¸ð üÐ ÝÐÐ" Ð¾ø##^L ý# £Ð
ÿÿÿÐÆÐÂ" Ð¾ø#^L ý ;Ý"üÆ ;¸ð üÐ ÝÐÐ"ü¶Æ^L ##^L # £Ð ÿÿÿÐÆÐÂ"ü¶Æ^L #
;ñþÐð;öýÿÆ

Ð Ð2HÐ>g^l Ð(Ð;I| Pairs ó âÐ ÿÿÿÐÆÐÂÝ"ÐÁÐ◄ ;ððÿ Æ ^aU^aU^aU^aU_q-Ð,Ð◄ ÐÁÐ-
Ð,Ð¶ ÐÁÐ◄ ÐÁÐÐ,Ð-Ð,Ð¶ Ð ÝÐÐ ÿÿÿÿÿÿÿ"Ð,Ð¶ Æ ##^{1/4}##- Æ##^{1/4}##^aU^aU^aU^aU^ñ £Ð
ÿÿÿÐÆÐÂÑ #Æ #^{1/4}#- Æ#^{1/4}# ; âÝ"Ð- â ;ðð
ÆÐ ÝÐÐ"Ð"Ç- Ü##Q##Ë\$##-# £Ð ÿÿÿÐÆÐÂ#- Ü#Q#Ë\$#- ;Ý"ÐÁÐ' ;ðð ÆÐ
ÝÐÐ"ÐÁÔ\$# £Ð ÿÿÿÐÆÐÂ"ÐÁÔ\$;Ý"ÐËÐ! ;ðð ÆÐ ÝÐÐ"ÐËÐ# £Ð
ÿÿÿÐÆÐÂ"ÐËÐ ;Ý"Ð,Ð- ;ðð ÆÐ ÝÐÐ"Ð,Ú\$# £Ð ÿÿÿÐÆÐÂ"Ð,Ú\$;Ý"Ð"Ú ;ðð ÆÐ
ÝÐÐ"Ð- ÐýÍ\$# £Ð ÿÿÿÐÆÐÂ"Ð- ÐýÍ\$;Ý"Ð" Æ ;ðð ÆÐ ÝÐÐ"Ð- ÚÍ\$# £Ð
ÿÿÿÐÆÐÂ"Ð- ÚÍ\$;Ý"Ð"Ðý ;ðð ÆÐ ÝÐÐ"Ð- Ð- Ì\$# £Ð
ÿÿÿÐÆÐÂ"Ð- Ð- Ì\$; ç çÝ"Ð^aÐ- ;ðð ÆÐ Ýðð"ÐÂÔJ:# £Ð ÿÿÿÐÆÐÂ"ÐÄ"J: ;Ý"ÐæÐ-
;ðð ÆÐ ÝÐÐ"ÐüÐeÈ##⁵/₄# £Ð ÿÿÿÐÆÐÂ"ÐüÐeÈ##⁵/₄ ;Ý"Ð^aÐ- ;ðð ÆÐ
Ýðð"Ð~ÐI'=# £Ð ÿÿÿÐÆÐÂ"Ð}ÐH'= ; â;ñþÐð;öýÿÆÐ Ð¶ ãÐ ó,
^l- Symbol^l ^l (Ð âÐÛ ó ç â;ñþÐð;öýÿÆÐ Ð2£Ð>±+^{3/4}-ÐÛ ó ç â;ñþÐð;öýÿÆÐ
õ£ÐÐ±(Æ^aÐÛ ó ç;ñþÐð;öýÿÆ/Ð ÐrÐÑÐ~Ðä^l ¶^l Ð+á~+ Variable Vectors ó;ñþÐð;öýð#Ð
Ð.mÐ\oe .(ÐÀñÝOutput
*þþFeature Map ó;ñþÐð;öý Æ Ð ÐçÐkÐÆÐ¥(Ð«ÐI Weights =
*þ| Cube
*þ Activities ó;ñþÐð;öýÿÆ7Ð ðÐ4^aÐ^a(;Ð5↑ Argument Positions ó;ñþÐð;öýÿü!! Ð
Ð\³ÐÖ(^{1/4}Ð]^l Cube ó ç Éÿ bÝ:ý@?D/Û#Ì²ËËÐ^l S¶ ¶ &- ÿ -ÿÿÿ^l ¶ Word

Microsoft Word

&- Œ ↑ Wordÿÿÿ Đ-- | Đđđđ Ý← ʹ | | ¶ đ | !! đ | | ¶ đ | | !! đ ³² @@\$\$Sie benötigen
Word 6.0c oder höher, ³² ÄĐ@Sum Macintosh-Grafiken anzuzeigen.
↳ -ĐWĐÀ◀ Đ;dpWORDĐWĐÀĐ ĐWĐÀ 4ĐVĐÊ84

ĐIDÆ8

- yyy4-DD.D°

! "D-D ; ñpDD,ý¶ | Times^L ¶
.↓ (D;J¶ Prolog's unification ó

Đô Đ-û-û

! "D-û ; ãpDD)\¶ infinite Unification ó

Đô Đ-ù-ù

! "D-ù ; ãpDD)_↑ linear Unification ó

Đô Đ-ĐT-ĐT

!"Đ-ĐT ;ñpĐĐ)dpour approach ó"Đ-Đ⁻ "Đ(Dú"Đ#Dú"Đ Dú"Đ^{3/4}Dú"Đ-
DÆ;ñpĐĐ(Đ1%[↓] 0,00 ó"Đ
Dú"ĐýDú"ĐđDú"ÆDú

Đô Đ↑ DĐ↑ Đ⁻

! " # \$ % & ' () * + , - . / : ; < = > ? @ [\] ^ _ ` { | } ~ ¡ ¢ £ ¤ ¥ ¦ § ¨ © ª « ¬ ® ¯ ° ± ² ³ ´ µ ¶ · ¸ ¹ º » ¼ ½ ¾

Đô ÆDÆĐ

! "ÆDÆ; ñpÐÐ(û | 20,00 ó"ÆDû"ÑDû"ÌDû"ÆDû

Ðó ÜDÜÐ

! "ÜDÆ; ñpÐÐ(à | 30,00 ó"Dû"¶Dû"±Dû"«Dû

Đô ÁDÁĐ

! "ÁDÆ;ñpÐÐ(Å | 40,00 ó";Dû"øDû"ñDû"êDû

Đô ĐĐĐ

! "DÆ; ñpÐÐ(ª | 50,00 ó"ÖDû"ÄDû" {Dû"uDû

Đô ãDãĐ

! "ãDÆ; ñþÐÐ(è | 60,00 ó"jDû"eDû" `Dû"ZDû

Đô pDpĐ⁻

! "pDÆ;ñpĐĐ(t | 70,00 ó"ODû"JDû"DDû"?Dû

Đô UDƯĐ

!"UDÆ;ñpĐĐ(Y | 80,00 ó"4Dû"/Dû")Dû"\$Dû

Đô :D:Đ

!":DÆ;ñpÐÐ(> | 90,00 ó"-DÆ;ñpÐÐ("← - 100,00 ó
à"à"à"à"4-cÐ.Ä84ÆæÐ.Ú84Ð,Ð³Ð.Ð584Ð,ÐtÐ.Ðê8;ñpÐÐ) Ð ó Ð-Ð⁻ -Ð⁻ -D-Ð⁻ Ð-DÐ-
Ð⁻ Ð,E E 4

a Ç;ñpÐÐ(³c- 100,00 ó4ß;ÏÛ;ñpÐÐ+^Ñ| 22,67 ó4Ð Ð Ð*Ð3;ñpÐÐ+]=^l 0,45 ó4Ð Ðw
Ð*Ðé;ñpÐÐ)[^l 0,45 ó;ñpÐÐ)† Ð ó;ñpÐÐ)^L Ð óy!4† -\oe ◀ à;±²á;^L ÿÆ
ÐÐ† ðÐyyyÆyy
yyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyLðyßÐ=ÐÆ ËË-\oe ◀ à;±²á;^L ÿÆ ÐÐð p◀ Ð;d

↑ ýĐKǫ ↓ ,@Ûÿđ,ýStandardþ| ℒ ¶ hĐ½Æ]ℒ aÝ↓ c¾&Đ`Đđ&
Überschrift 1đĐÝUÅ]| ^Đ(đ`Đđ(
Überschrift 2| đ½x| UÅ]| &A@Úÿ; &³Absatz-Standardschriftart\$`ĐÚ\$
Kopfzeileþǫ ǫ ýđ"↓ o#ĐđBÆOĐđB+ Literaturhinweis-
+ | ℒ ◀ 7đ!! ...Æ¶ hĐ½Èǫ Þℒ 7đ ¼9!đℒ]ℒ -ÆOĐ↑ Đ-ýAbstractý◀+ \$đ◀ 0đ- N| - Q| (Q
ÿÿĐ ÿÿđ ÿÿ ÿÿ↓ ↓ ÿÿ ÿÿ- ↓ ÿÿÝ ÿÿý ÿÿ ÿÿ
ÆÝJ+ O← à"^^%"/ø7?<OF- NÊđĐ ↓ đđ,đ£ĐBĐ- KÝûya ýℒ UùOℒ v-Æ-Ú-7-
D<- QPQRSTUw²û² ²MctCvC- N!! :ÿÿiÄ!! :ÿÿiÄ[◀ Office 4.2
GermanFSekretariat_NKI:Sekretariat:Internet:wina.www:Papers:UnifikationWCES94ÿ@ÄĐ
ÝĐĐ¼ĐPđ- ℒ \½êĐMTimes New RomanþđSymbolÞ- MArialÞêĐMTimes

