Integrating Learning and Inference in Multi-Agent Systems Using Cognitive Context

Bruce Edmonds and Emma Norling

Centre for Policy Modelling
Manchester Metropolitan University
bruce@edmonds.name,norling@acm.org

Abstract. Both learning and reasoning are important aspects of intelligence. However they are rarely integrated within a single agent. Here it is suggested that imprecise learning and crisp reasoning may be coherently combined via the cognitive context. The identification of the current context is done using an imprecise learning mechanism, whilst the contents of a context are crisp models that may be usefully reasoned about. This also helps deal with situations of logical under- and over-determination because the scope of the context can be adjusted to include more or less knowledge into the reasoning process. An example model is exhibited where an agent learns and acts in an artificial stock market.

1 About Context

Both learning and reasoning are far more feasible when their scope is restricted to a particular context because this means that only the relevant knowledge needs to be dealt with. However if any significant degree of generality is to be obtained in this manner [1] then an intelligence must be able to appropriately change this focus as the external context (the context we inhabit in [2]) changes. In other words there needs to be some internal correlate of the external context that allows an intelligence to identify which set of beliefs apply. We will call this internal correlate the cognitive context (this is the "internal" approach identified in [3]). There are (at least) two tasks necessary for this:

- identifying the appropriate cognitive context from perceptions, and
- accessing the appropriate beliefs given the identified cognitive context.

The success of this strategy of assessing the relevance of knowledge via identifiable "contexts" depends upon whether the environment is usefully divided up in such a manner. This is a contingent matter – one can imagine (or devise) environments where this is so and others where it is not. The "pragmatic roots" of context, i.e. why context works, depends upon the underlying pattern of commonalities that occur in an environment or problem domain [4]. A cognitive context indicates the boundaries of what might be relevant in any situation.

Context serves not only to make it feasible to deal with our knowledge at any one time but also, at a more fundamental level, to make our modeling of

the world at all feasible. The efficacy of our limited learning and inference in dealing with our complex world is dependent on the presumption that many of the possible causes or effects of important events remain relatively constant [5]. Otherwise we would need to include all possible causes and affects in our models and decision making processes, which is clearly infeasible. It is the existence of relative constancy of many factors in particular situations that makes our limited modeling ability useful: we can learn a simple model in one circumstance and successfully use it in another circumstance that is sufficiently similar to the first (i.e. in the same "context").

It is the possibility of the transference of knowledge via fairly simple models from the circumstances where they are learnt to the circumstances in which they are applied that allows the emergence of context. The utility of "context" comes from the possibility of such transference. If this were not feasible then "context", as such, would not arise. For such a transference to be possible a number of conditions need to be met, namely that:

- some of the possible factors relevant to important events are separable in a practical way,
- a useful distinction can be made between those factors that can be categorized as foreground features and the others (the constant, background features),
- similar background factors are capable of being reliably recognized later on as the same "context",
- the world is regular enough for such models to be learnable,
- the world is regular enough for such learnt models to be useful where such a context can be recognized.

While this transference of learnt models to applicable situations is the basic process, observers and analysts of this process might identify some of these combinations of features that allow recognition and abstract them as "a context". Note that it is not necessarily possible that such an observer will be able to do this as the underlying recognition mechanism may be obscure, too complex or difficult to analyze into definable cases.

Such a strategy answers those of the "frame problem" [6]. Firstly, although the frame problem may be unsolvable in general it is learnable in particular contingent cases. Secondly, the identification of appropriate contexts are not completely accessible to reasoning or crisp definition – rather it is an unreliable, information-rich, and imprecise process. Thus knowing B in context A, is not translatable into statements like $A \rightarrow B$, because the A is not a reified entity that can be reasoned about.

The power of context seems to come from this combination of "fuzzy," fluid context identity and crisp, relatively simple context "contents". Thus context straddles the fields of Machine Learning and Artificial Intelligence. Machine learning seems to have developed appropriate methods for complex and uncertain pattern recognition suitable for the identification of context. Artificial Intelligence has developed techniques for the manipulation of crisp formal expres-

sions. Context (as conceived here) allows both to be used for different functions in an coherent way.

2 The Research Context

In 1971, in his ACM Turing Award lecture, John McCarthy suggested that the explicit representation and manipulation of context might be a solution to the effective lack of generality in many AI systems (these ideas were later developed and written up in [1]). Since then context and context-like ideas have been investigated in both the AI and ML communities, culminating in several workshops and a series of international conferences entirely devoted to the subject. Below work in these areas is briefly summarized in order to set the stage for the work that is reported here.

2.1 Context in Reasoning

McCarthy's idea was to reify the context to a set of terms, i, and introduce an operator, ist, which basically asserts that a statement, p, holds in a context labeled by i. Thus:

$$c : ist(i, p)$$

reads ""p is true in context i" which is itself asserted in an outer context c. ist is similar to a modal operator but the context labels are terms of the language. Reasoning within a single context operates in a familiar way, thus we have:

$$\forall i(ist(i, p) \wedge ist(i, p \rightarrow q) \rightarrow ist(i, q))$$

In addition one needs a series of 'lifting' axioms, which specify the relation between truth in the different contexts. For example if $i \geq j$ means that "i is more general than context j", then we can lift a fact to a supercontext using:

$$\forall i \forall j(i \geq j) \wedge (ist(i, p) \wedge \neg ab(i, j, p) \rightarrow ist(j, p))$$

where ab is an abnormality predicate for lifting to supercontexts. This framework is developed in [7]. There are a whole series of formal systems which are closely related to the above structure, including, notably: the situations of Barwise and Perry [2], Gabbay's fibered semantics [8], and the local semantics of the Mechanized Reasoning Group at Trento [9].

One of the problems with this sort of approach is that it is likely that trying to apply generic reasoning methods to context-dependent propositions and models, will be either inefficient or inadequate [10]. The generic approach forces a choice of the appropriate level of detail to be included, so that it is likely that either much information that is irrelevant to the appropriate context will be included (making the deduction less efficient) or much useful information that is specific to the relevant context may be omitted (and hence some deductions will not be possible).

Another problem is that, in practice, this type of approach requires a huge amount of information to be explicitly specified: contexts, contents of each context and bridging rules.

2.2 Context in Learning

The use of context in machine learning can be broadly categorized by goal, namely: to maintain learning when there is a hidden/unexpected change in context; to apply learning gained in one context to different context; and to utilize already known information about contexts to improve learning. There are only a few papers that touch on the problem of learning the appropriate contexts themselves. Included in those that do, Widmer [11] applies a meta-learning process to a basic incremental learning neural net; the meta-algorithm adjusts the window over which the basic learning process works. Here it is an assumption that contexts are contiguous in time and so a time-window is a sufficient representation of context. Harries et al. [12] employ a batch learner as a meta-algorithm to identify stable contexts and their concepts; this makes the assumption that the contexts are contiguous in the "environmental variables" and the technique can only be done off-line. Aha describes an incremental instance-based learning technique which uses a clustering algorithm to determine the weight of features and hence implicitly adjust context [13].

Contextual knowledge has been used to augment existing machine learning techniques in a number of instances. Turney [14] used explicit identification of what the contextual factors would be, but others have used implicit features (e.g. Aha [13]). Turney [15] discusses the problem of the effects of context on machine learning and surveys some heuristics used to mitigate these effects [16].

2.3 Context in Human Cognition

The use of context is a pervasive heuristic in human cognition. It appears that we use context in almost every area of our thinking and action, including: language understanding; memory; concepts and categorization; affect and social cognition and (probably) problem solving and reasoning [17]. In the past some researchers perceived the context-dependency of human thought purely as a disadvantage or side-effect, but now it is becoming increasingly clear that it is an essential tool for enabling effective learning, reasoning and communication in a complex world. We speculate that much of the power of human thought comes from an ability to rapidly switch between different cognitive contexts, whilst only doing relatively simple inference within these (with the exception of problem solving or fault-diagnosis).

It has been recognized for a while that the external (and linguistic) context plays a role in the understanding of natural language. However it is only recently that the importance of context in communication has been appreciated. The external context is not merely a resource for understanding utterances that is accessed when all other mechanisms fail; a way of sorting out otherwise ambiguous sentences. Rather it is one of the primary mechanisms. We will not discuss the use of context in language more as it is a huge and controversial area in which we are not experts.

Although human cognition is not a necessary starting point for motivating the design of an intelligence, it is a fruitful one, especially when looking for solutions

that will scale up to cope with problems of real world complexity. What the human case shows is that it does appear that context is a useful heuristic in at least some real-world cases.

3 Combining Context-Dependent Learning and Reasoning

Restricting both reasoning and learning to an appropriate context makes both more feasible. However, as with any other technique, there are a number of difficulties with applying a context-dependent approach to reasoning. Firstly:

- Explicitly specifying a set of knowledge appropriate for a whole set of potential contexts is both time-consuming and labor-intensive.

Thus with a few honorable exceptions (e.g. CYC [18]), most systems of context-dependent learning or reasoning are only tried out with a few contexts. A possible answer to this (and the one employed here) is to learn the contexts and the context-dependent knowledge. The second is easier than the first; for, as indicated above in Sect. 2.2, there are a number of techniques to learn the knowledge associated with contexts.

The learning of the contexts themselves (i.e. how to recognize when a set of beliefs learnt in a previous situation are again applicable) requires a sort of meta-learning. As documented above, there are such techniques in existence. However most of these either require reasonably strong assumptions about the particular nature of the contexts concerned. An exception is [19] which describes how contexts can be co-learnt along with the knowledge associated with those contexts. This applies an evolutionary learning algorithm where the knowledge is distributed across a space, where different positions in that space are associated with different set of perceptions or different parts of a problem. This can be clearly understood via the following ecological analogy. If the space can be thought of as a landscape where different parts of the landscape have different properties, and different plants require different properties (some might thrive in marshy land, others sunny and dry etc.). The set of solutions can be seen as varieties of a plant. The different varieties propagate and cross with others in each locality so that, eventually, each variety adapts and, at the same time, spreads across the areas that it is best adapted for. The patches where different sets of varieties thrive define the different ecological niches – corresponding to the different contexts via this analogy.

The ability to learn context allows us to progress beyond the 'loose' loop of:

```
repeat
   learn/update beliefs
   deduce intentions, plans and actions
until finished
```

to a more integrated loop of:

```
repeat
   repeat
      recognise/learn/choose context
      induce/adapt/update beliefs in that context
      deduce predictions/conclusions in that context
   until predictions are consistent
         and actions/plans can be determined
   plan & act
until finished.
```

Such a co-development of cognitive contexts along side their "contents" gives rise to a new problem when the knowledge in these contexts is used to infer predictions and decisions. Thus a second problem is this:

– When some of the contents turn out to be wrong, how can one tell when it is the context that is wrong and when it is the contents that are wrong?

There is no universal answer to such a question – it will, in general, depend upon the nature of the domain and hence the appropriate contexts in that domain. However there is a heuristic, as follows: if only a few of the elements of knowledge associated with a context are disconfirmed, it is likely that these are wrong (update the set); if many of the elements are disconfirmed then it is likely that the context is wrong (change it and learn from this).

Thus in the proposed architecture there are four modules: (1) the context identification system; (2) the context-dependent memory; (3) the local learning/induction algorithm; and (4) the inference system, as shown in Fig 1.

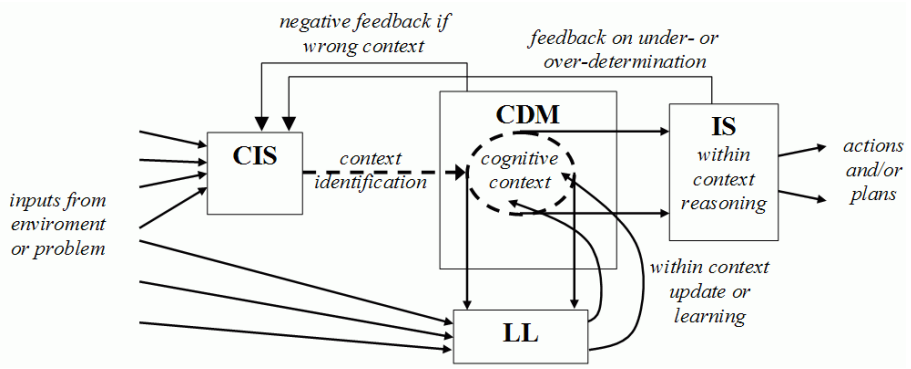

Fig. 1. How the context-identification system (CIS), the context-dependent memory (CDM), the local learning algorithm (LL), and Inference system (IS) work together.

The context identification system (CIS) takes a rich range of inputs and learns in a flexible and imprecise way an indication of the context (which it outputs to

the memory). The CIS learns as the result of negative feedback when too much of the knowledge in the cognitive context is simultaneously disconfirmed.

The context-dependent memory (CDM) takes the indication given by the CIS and identifies all those memory items stored within that context. It evaluates the (current) truth of these and if too many are false it returns negative feedback to the CIS which will identify another context. If a sufficient number of indicated contents are true, then the local learning updates the items within that context. Those items that are (currently) true are passed to the inference system.

The local learning algorithm (LL) performs a local update of the knowledge in the memory. It may include the propagation of successful items towards the focus, but may also include the deletion/correction of items that were false and the possible insertion of new induced/learned.

Finally the planning/inference system (IS) tries to deduce some decisions as to the actions or plans to execute. It could do this in a number of ways, but this could include trying to predict the future states of the world given possible actions and comparing the predictions using its goals.

Two common problems with inference systems that attempt to deduce predictions or decisions from an arbitrary collection of knowledge are under- and over-determination. Under-determination is when there is not enough information to come to a conclusion or decision that needs to be reached. In other words there may be a key proposition, α, such that neither α nor $\neg\alpha$ can be inferred. Over-determination is when there is contradictory information in the knowledge, i.e. when there is an α such that both α and $\neg\alpha$ can be deduced.

This architecture allows a useful response in these two situations. In the case of under-determination the context can be expanded so that more knowledge can be made available to the IS so that it may make more inferences. In the case of over-determination the context can be reduced so that some of the knowledge can be excluded, the knowledge that is peripheral to the context.

Many non-monotonic logics can be seen as attempts to solve the above problems in a generic way, i.e. without reference to any contingent properties obtained from the particular contexts they are applied in. So, for example, some use entrenchment to determine which extra information can be employed (e.g. oldest information is more reliable [20]), and others allow a variety of default information to be used (e.g. using extra negative knowledge as long as it is consistent [21]). These may work well on occasion, but they can not exploit any of the relevance relations specific to the particular knowledge and context.

4 A Demonstration Model

To show these ideas working in practice a demonstration model is briefly exhibited. This deploys particular algorithms to the processes of context identification, local learning/update and inference. The authors suspect that there are far better techniques available for these. However, the purpose here is to show how these processes can be usefully integrated using context dependency, and is thus

not too concerned with the exact tuning of the parts. For more detail on the particulars of the model, see the appendix to this paper.

4.1 The Environment

To support contextual learning, the chosen environment needs to be sufficiently rich (1) to be worth learning contextual information about, and (2) that an entity may predict and reason about many different aspects at once. The heuristic must be able to distinguish between an incorrect choice of context (when many beliefs/predictions are proven to be wrong) and incorrect beliefs/predictions (when only a few are wrong). The environment selected to meet these requirements was a small artificial stock market [22], where the past actions of other traders, the prices, volumes, money etc. are observable. This is a constrained but dynamic setting, where any actions have significant effects on the environment because the other traders will then adapt their strategies. The prices follow a pattern typical of many markets – there are booms and busts; much apparent noise; only a weak, long-term correlation of prices with dividends; and a long-term exponential trend in prices.

4.2 The Context Identification System (CIS)

The CIS is implemented using a simple table with four entries: a vector of possible perceptions; a weight; a radius; and an output coordinate of the CDM. The distance between the vector of input perceptions and the first column entries multiplied by the weight are calculated and the row with the minimum value fired. The radius and coordinate are output to the CDM. In the case of negative feedback, the fired weight is increased. Thus the table forms a simple self-organized critical system [23].

4.3 The Context-Dependent Memory (CDM)

The CDM is another table, associating items of knowledge expressed as strongly typed trees, with coordinates. When given a focus coordinate and radius from the CIS it identifies those items within the radius of the focus coordinate. This CDM thus has similarities to the way that SDM memory works [24]. It evaluates these knowledge items against the current state of the world and the proportion of correct items calculated. If this value is low, negative feedback is provided to the CIS and control is passed back to the CIS, otherwise the LL module acts on the identified knowledge set and (in parallel) the set is passed to the IS.

Knowledge items are represented as typed trees using a fairly rich language of nodes and terminals. These include nodes for logical operators (and, or, implies, not); arithmetic operators (plus, minus, times, divide); comparisons (greater than, less than); temporal operators (last, variable lag) and others (e.g. randomIntegerUpTo). Terminals include: Boolean constants (T, F); a selection of numeric constants; prices of stocks; actions of other traders in each stock; cash owned; stocks owned; trading volumes; stock index; whether the price of a stock has gone up or down; etc.

4.4 The Local Learning Algorithm (LL)

The LL is based on Strongly-Typed Genetic Programming [25]. Random selections of the currently true items in the context are propagated, mutated and crossed and placed towards the focus coordinate. A proportion of incorrect items are culled. To stop the memory growing too big, if the total number of items is larger than a set size then items in the whole memory that have not been accessed for the longest time are also culled.

4.5 The Inference System (IS)

The inference system takes the set of items provided by the CDM (that are currently true) and evaluates them (as far as possible) from the point of view of the hypothetical next time instance – this simplifies them somewhat. It then repeatedly tries a set of inference rules (MP, etc.). The database is then examined to see if: (A) false has been inferred, in which case the current radius (in fired row of CIS and CDM) is reduced; (B) it has inferred that the price of any stocks will increase or decrease (at the next time instant), if so (without contradictory predictions) the appropriate buy or sell action is taken. (If the price movement cannot be inferred, the current radius is increased.)

4.6 Preliminary Results

The agent that used the architecture did not initially do as well as the GP learners it was pitted against. The latter learn in a quick-and-dirty way evolving strategies that would have performed best over the last 5 time periods only. The context agent has a lot more learning to do than them before it becomes effective. As you can see from Fig. 2 the context trader does worse up to time 400 (as it learns) but then catches up by roughly 40 orders of magnitude.

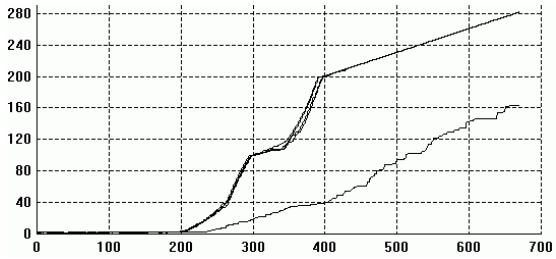

Fig. 2. The logarithm of the total assets of traders (context is lower line) against time.

However we can show that this agent does learn and reason in a context-dependent way. Figure 3 shows that the agent does make trades and hence infers enough to make predictions about the movements of prices.

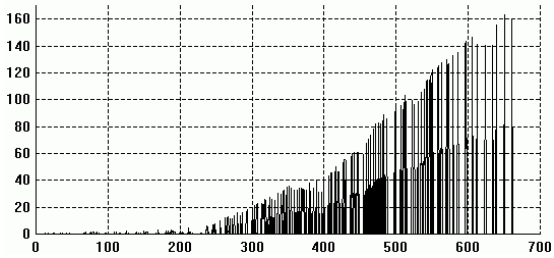

Fig. 3. The logarithm of the values of trades made by the context agent over time.

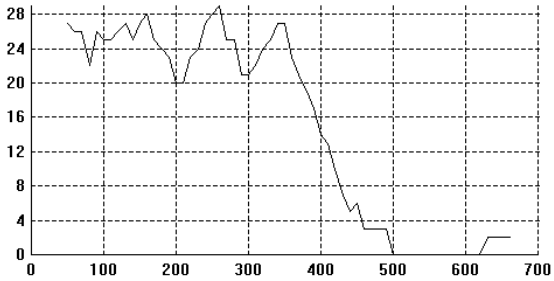

Fig. 4. The number of bad contexts identified over the last 50 time periods.

The last graph (Fig. 4) shows the rate of bad context identification over time – this drops to zero over the time.

Clearly these results are only from a single run and thus do not prove anything. However they are not to prove the technique but merely show its feasibility. In the future we hope to be able to greatly improve these results.

5 Conclusion

We have shown how context can be used to integrate learning and reasoning in a coherent manner, improving the tractability of both whist retaining most of the advantages of both. As an additional benefit it provides sensible responses to the problems of under- and over-determination in knowledge. Knowledge update can be done in a specific manner reflecting the properties and relevance of the knowledge items rather than relying on generic heuristics.

This is but one way in which learning and reasoning can interact. Due to the divide between the ML and agent communities, there are too few studies of the possible interactions between learning and reasoning processes. These interactions are important because they can result in outcomes that are not obvious from a simple knowledge of the components. This study goes a little way in this direction.

Acknowledgments

We would like to thank the participants of the conferences on Modelling and Using Context for their comments on these and related ideas, particularly Anita Fetzner.

Appendix – A More Detailed Specification

Here we give more details about the structure and working of the modules and simulation as summarised above in Sections 3 and 4. The overall structure is summarised in Section 3 above and illustrated in Figure 1, here we concentrate upon the particular instantiation of the four modules used in the demonstration model as well as the test environment in which the integrated agent was tested. It must be emphasised that the particular algorithms we used are somewhat arbitrary there are probably far better ones available. The purpose of this model is to be a demonstrator model, not to represent an ideal. We start with the four modules and then deal with the environment.

General Structure of Agent

The general structure is summarised in Section 3 above and illustrated in Figure 1. Key signals into/out from/between modules are as follows:

- A vector of 69 different perceptions are fed into both the CIS and LL;
- The (guess at the) current appropriate context is sent from CIS to CDM as a pair of a coordinate and a radius;
- If the context is wrong a bad context signal is sent from CMD to CIS
- If the set of expressions passes basic tests, the set of expressions within the currently indentified context is sent to the IS;
- If the set of expressions is inconsistent a signal of over-determination is sent to the CIS, if there is insufficient information to deduce a decision then a signal of under-determination is sent to the CIS;
- The IS outputs decisions in terms of buying and selling stocks;
- The LL acts upon the contents of the CDM using present and past information about the environment to do so.

Perceptions and Actions

The agent continually perceives its environment in a rich way. In particular it compiles the following information: `doneByLast` for each other contact and stock; `IDidLastTime`, `averageOccuredToStockLast`, `dividendOf`, `ISoldLastTime`, `IBoughtLastTime`, `presentStockOf`, `priceDownOf`, `priceNow`, `priceUpOf`, `priceLastWeek` for each stock; `averageDoneByLast` for each contact; and one each of `indexLastTime`, `indexNow`, `myMoney`, `randomPrediction`, `randomBoolean`, `T`, `F`, `totalStockValue`, `volumeLastTime`,

`onAvIBoughtLastTime`, `maxHistoricalPrice`, `onAvISoldLastTime`; and finally `averageIndexOverLast`, `indexLag`, `indexTrendOverLast` for each of time lags 1..5. This makes up a vector of 69 given the settings in the current model. The actions are simply how much to buy or sell of each stock. There are 4 other traders (but only 3 of these are contacts) and 3 stocks.

Context-identification system (CIS)

This was implemented as a simple lookup table, with each row representing a potential context that could be recognised. The first column is a vector of values, one for each of the different perceptions of the agent. Basically, given a particular vector of current perceptions of the environment, the closest in the table is identified. The distance measure is a cartesian distance between the two vectors but multiplied by the number in the second column, the weight. This weight allows for the generality of the context to be adjusted – the lower the weight the more general are the conditions under which the context is indentified. The last two columns are the radius and the memory coordinates that are output to the CDM when the row is fired.

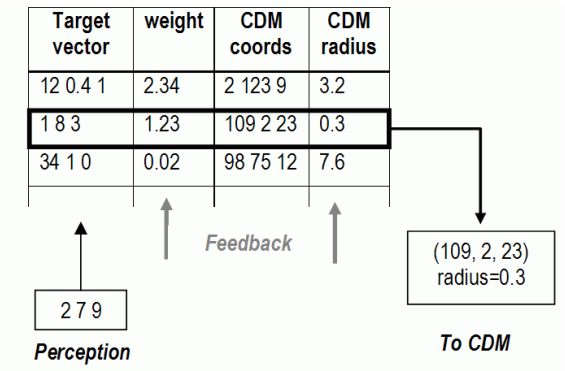

Fig. 5. The CIS implemented as a simple table using the closest match moderated by the weights. The corresponding CDM coordinate and radius is output.

The table was randomly initialised; the input vector in the first column is of size 69 initialised with random values in the [-10, 110] range; the CDM coordinates being in the [0,100] range; the intial weights were all 1; the initial radii were all 20.

Changes occur to the CIS table as follows. If the set of knowledge output from the CDM turns out to be inconsistent or inadequate, the IS signals this to the CIS and the radius in the fired row is reduced or increased respectively. If the context was wrongly identified then the corresponding weight is increased to reduce the times this is fired.

Parameters are: number of rows=100; weight increase factor on 'bad context' signal=1.5; expand/contract factor on radius on under-/over-determination signal=1.1; initial weight = 1; initial radius=20.

Context-dependent memory (CDM)

The CDM is a multi-dimensional space (but of fewer dimensions than that of the perception space in the CIS, in this case 3). Scattered across this space are items of knowledge expressed as GP-like expressions. When given a coordinate and a radius the CDM retrieves the set of all expressions within the given radius of the given coordinate. It then tests these against the current state of the world (as perceived) if the proportion of these falls below a critical value, this indicates that this is the wrong context so control is passed back to the CIS and the corresponding wieght is increased. This critical value starts low and is gradually ramped up.

Initially the memory is randomly seeded with random expressions to a set depth. These are then gradually evolved by the LL module as context-dependent learning occurs. The language of these typed trees includes the following nodes:
`AND`, `averageDoneByLast`, `averageIndexOverLast`, `averageOccuredToStockLast`, `averageSaidBy`, `averageSaidByLast`, `boundedByMaxPrice`, `divide`, `dividendOf`, `doneByLast`, `greaterThan`, `IBoughtLastTime`, `IDidLastTime`, `indexLag`, `indexTrendOverLast`, `ISaidLastTime`, `ISaidYesterday`, `ISoldLastTime`, `lagBoolean`, `lagNumeric`, `lastBoolean`, `lastNumeric`, `lessThan`, `minus`, `NOT`, `OR`, `plus`, `presentStockOf`, `priceDownOf`, `priceLastWeek`, `priceNow`, `priceUpOf`, `randomIntegerUpTo`, `saidBy`, `saidByLast`, `statResult`, `talkResult`, `times`;
and terminals:
`onAvISoldLastTime`, `maxHistoricalPrice`, `avTraderPredictedIndex`, `F`, `onAvIBoughtLastTime`, `indexLastTime`, `randomPrediction`, `randomBoolean`, `totalStockValue`, `T`, `myMoney`, `volumeLastTime`
plus a slection of numeric constants, the names of the traders and the stocks. Thus an example expression is:

```
[IMPLIES [greaterThan [times [doneByLast ['gPTrader-4'] ['stock-3']]
[presentStockOf ['stock-2']]] [talkResult ['stock-2']]] [lastBoolean
[priceDownOf ['stock-1']]]]
```

Parameters are: coordinate dimension=3; coordinate range=[0, 100]; initial depth of trees=4; number of numeric constants=10; range of numeric constants=[0, 50]; initial population expressions=1000; maximum number of expressions=4000.

Local learning algorithm (LL)

The LL algorithm samples the contents of the CDM and applies an evolutionary algorithm to it. Each cycle the LL does the following:

```
For each expression in the current context currently true
    If randomNumber < propogationProb
        Copy expression 25% towards the focus
```

```
            Mutating it with probability mutationProb
    If anotherRandomNumber < crossoverProb
        Randomly choose anotherExpression in context
        Cross expression and anotherExpression and store
            Store result shifted 25% towards the focus
            From the average position of parents
Next expression
For each expression in the current context currently false
    If randomNumber < cullProb
        Delete expression
Next expression
```

If the total population is greater than a set maximum, then those expressions that have been accessed least recently in the whole memory are deleted to bring the population down to the maximum.

The parameters for this module are: crossoverProb=0.1; cullProp=0.667; mutationProp=0.05; and propagationProp=0.4; maximum population=4000.

Inference system (IS)

The IS does the following: it takes the set of expressions in the current context that are currently true; it simplifies the expressions using the current state of the perceptions (substituting the known values for subexpressions and doing some arithmentic and logical simplification); then it applies a set of inference rules to the resulting set for a maximum of 10 sub-cycles (or until no new inferences can be made); the result of this (in terms of inferences about the next price movements) determines the actions or signals that result.

Inference rules used are the following: $A \wedge B \models A$; $A \wedge B \models B$; $A, A \rightarrow B \models B$; $\neg B, A \rightarrow B \models \neg A$; $\neg\neg A \Rightarrow A$; $\neg(priceUp) \Rightarrow priceDown, \neg(priceDown) \Rightarrow priceUp$.

The results are evaluated as follows: if neither priceUp nor priceDown are inferred for the next time period for any stock then no action is taken and a signal sent that this context is under-determined; if for a stock priceUp is inferred but not priceDown for that stock the agent buys; if for a stock priceDown is inferred but not priceUp for that stock the agent sells; and if both priceUp and priceDown are inferred the over-determined signal is sent.

The parameters for this module are: maximum number of inference cycles=10.

Environment

The environment is a stock market, with a fixed number of stocks and traders, plus a single market maker who sells and buys to and from the traders. There is a fundermental for each stock, which is a dividend rate. This determines the rate at which a stock yeilds income from owning it. These rates change slowly in a random-walk fashion. Stock prices are changed by the market maker depending upon the balance of supply and demand. There is a limited amount of stocks

in the system, but the market maker has an unlimited amount of cash (it can always afford to buy and it never goes broke).

All traders are initialised with a certain amount of cash and a small random level of each stock. There is a running-in period where all actions are random, to provide for the initial conditions for learning. If the demand was greater than the supply then the price goes up by the set increment, otherwise it goes down by the same increment.

The parameters for the market maker and environment are: num GP traders= 4; number of stocks=3; initial money=20; the initial dividend is randomly selected from the range [0,0.02] for each stock; the initial price is randomly selected from the range [4, 6] for each stock; the initial stock for each agent is randomly selected from range [2, 5] for each stock; number of contacts each trader has = 3; the factor prices are incremented or decremented = 1.04; running-in time cycles =10.

5.1 GP Traders

The other traders that the context trader is competing against have identical abilities in terms of perception and action but have a much simpler (non context-dependent) GP algorithm. Basically each cycle its population of strategies is evolved once using a fairly standard GP algorithm with the fitness determined by an evaluation of the increase in total assets that would have resulted if that strategy was used over a certain number of time periods. The best is used to determine the current strategy which is then moderated for possibility (e.g. it may want to buy 2000 units of a certain stock but only have enough cash for 5 units!).

Each strategy is composed of a set of typed trees which, when evaluated, yields a number. There is one tree for each stock, plus one stats tree which can be used to learn a useful result that can be used in the other trees. Each GP trader agent has a small population of these strategies. The language for these trees is the same as that used in the context agent.

Parameters for these agents are: number of models = 30; initial depth of trees = 5; type of the root of the tree = numeric; time horizon for fitness evaluation = 5 cycles; number of stats = 1; crossover probability = 0.8; probability of a new (randomly generated) strategy 0.05.

References

1. McCarthy, J.: Generality in artificial-intelligence – turing award lecture. Communications of the ACM **30**(12) (1987) 1030–1035
2. Barwise, J., Perry, J.: Situations and Attitudes. MIT Press, Cambridge (1983)
3. Hayes, P.: Contexts in context. In: Context in Knowledge Representation and Natural Language, AAAI Fall Symposium, AAAI Press (1997)
4. Edmonds, B.: The pragmatic roots of context. In Bouquet, P., Serafini, L., Brezillon, P., Benerecetti, M., Castellani, F., eds.: Modeling and Using Contexts: Proceedings of the 2nd International and Interdisciplinary Conference. Number 1688 in LNAI, Springer (1999) 119–134

5. Zadrozny, W.: A pragmatic approach to context. In: Context in Knowledge Representation and Natural Language, AAAI Fall Symposium, AAAI Press (1997)
6. McCarthy, J., Hayes, P.: Some philosophical problems from the standpoint of ai. Machine Intelligence **4** (1969) 463–502
7. McCarthy, J., Buvac, S.: Formalizing context (expanded notes). In Aliseda, A., van Glabbeek, R., Westerstahl, D., eds.: Computing Natural Language. Number 81 in CSLI Lecture Notes. CSLI Publications, Stanford, California (1998) 13–50
8. Gabbay, D.: Fibring logics. Clarendon, Oxford (1999)
9. Ghidini, C., Giunchiglia, F.: Local models semantics, or contextual reasoning = locality + compatibility. Artificial Intelligence **127**(3) (2001) 221–259
10. Greiner, R., Darken, C., Santoso, N.: Efficient reasoning. ACM Computing Surveys **33**(1) (2001) 1–30
11. Widmer, G.: Tracking context changes through meta-learning. Machine Learning **27** (1997) 259–286
12. Harries, M., Sammut, C., Horn, K.: Extracting hidden contexts. Machine Learning **32** (1998) 101–112
13. Aha, D.: Incremental, instance-based learning of independent and graded concept descriptions. In: Proceedings of the 6th International Workshop on Machine Learning, Morgan Kaufmann (1989) 387–391
14. Turney, P.: Robust classification with context-sensitive features. In: Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, IEA/AIE-93, Edinburgh, Gordon and Breach (1993) 268–276
15. Turney, P.: The identification of context-sensitive features: A formal definition of context for concept learning. In: ICML-96 Workshop on Learning in Context-Sensitive Domains, Bari, Italy (1996) 53–59
16. Turney, P.: The management of context-sensitive features: A review of strategies. In: ICML-96 Workshop on Learning in Context-Sensitive Domains, Bari, Italy (1996) 60–66
17. Kokinov, B., Grinberg, M.: Simulating context effects in problem solving with ambr. In Akman, V., Bouquet, P., Thomason, R., Young, R., eds.: Modelling and Using Context. Number 2116 in LNAI, Springer (2001) 221–234
18. Lenat, D.: CYC – a large-scale investment in knowledge infrastructure. Communications of the ACM **38**(11) (1995) 33–38
19. Edmonds, B.: Learning and exploiting context in agents. In: Proceedings of the 1st International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS), Bologna, Italy, ACM Press (2002) 1231–1238
20. Gärdenfors, P.: Epistemic importance and minimal changes of belief. Australasian Journal of Philosophy **62**(2) (1984) 136–157
21. Reiter, R.: A logic for default reasoning. Artificial Intelligence **13** (80) 81–132
22. Palmer, R., Arthur, W.B., Holland, J.H., LeBaron, B., Taylor, P.: Artificial economic life – a simple model of a stock market. Physica D **75** (1994) 264–274
23. Bak, P.: How nature works : the science of self-organized criticality. Copernicus, New York (1996)
24. Kanerva, P.: Sparse Distributed Memory. MIT Press (1988)
25. Montana, D.J.: Strongly typed genetic programming. Evolutionary Computation **3**(2) (1995) 199–230