Travian Meta Alliances ⋆

Ingo Thon[1], Niels Landwehr[2], and Luc De Raedt[1]

[1]Department of Computer Science, Katholieke Universiteit Leuven, Celestijnenlaan 200A, 3001 Heverlee, Belgium
[2]Department of Computer Science, University of Potsdam, August-Bebel-Str. 89, 14482 Potsdam, Germany
ingo.thon@cs.kuleuven.be, landwehr@cs.uni-potsdam.de, luc.deraedt@cs.kuleuven.be

Abstract.

1 CPT-L

Let us first introduce some terminology. A logical *atom* is an expression of the form $p(t_1, \ldots, t_n)$ where p/n is a *predicate symbol* and the t_i are *terms*. Terms are built up from constants, variables, and functor symbols. Constants are denoted in lower case (such as a), variables in upper case (such as Z), and functors by f/k where k is the arity of functor f. The set of all atoms is called a *language* $\mathcal{L}$. *Ground* expressions do not contain variables. Ground atoms will be called *facts*. A substitution θ is a mapping from variables to terms, and $b\theta$ is the atom obtained from b by replacing variables with terms according to θ. As an example, consider the substitution $\theta = \{Z/a\}$ that replaces variable Z with a, as in $b\theta = p(a)$ for $b = p(Z)$.

Complex world states can now be described in terms of *interpretations*. An interpretation I is a set of ground facts $\{a_1, \ldots, a_N\}$. These ground facts can represent objects in the current world state, their properties, and any relationship between objects. As an example, consider the representation of the state of a multiplayer game in terms of an interpretation as depicted in Figure 1. The semantics of CPT-L is based on CP-logic, a probabilistic first-order logic that defines probability distributions over interpretations [1]. CP-logic has a strong focus on causality and constructive processes: an interpretation is incrementally constructed by a process that adds facts which are probabilistic *outcomes* of other already given facts (the *causes*). CPT-L combines the semantics of CP-logic with that of (first-order) Markov processes. Causal influences only stretch from I_t to I_{t+1} (Markov assumption), are identical for all time steps (stationarity), and all causes and outcomes are observable.

Definition 1. *A* **CPT-theory/model** *is a set of rules of the form*

$$r = (h_{1,1} \wedge \ldots \wedge h_{1,k_1} : p_1) \vee \ldots \vee (h_{n,1} \wedge \ldots \wedge h_{1,k_n} : p_n) \longleftarrow b_1, \ldots, b_m$$

where the $h_{i,j}$ are logical atoms, $p_i \in [0, 1]$ are probabilities s.t. $\sum_{i=1}^{n} p_i = 1$, and the b_l are literals (i.e., atoms or their negation).

⋆ This is an excerpt of a paper accepted for publication in the Machine Learning Journal Special issue on Mining and Learning with Graphs and Relations.

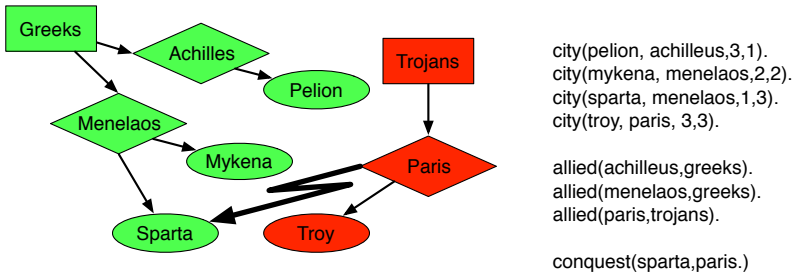

Fig. 1: Example for the state of a multiplayer game represented as a graph structure and, equivalently, as a logical interpretation. The rectangles in graphical representation refer to alliances, diamonds to players, and ellipsis to cities. The last two arguments of city in the logical representation refer to the location of the city.

A conjunction $h_{i,1} \wedge \ldots \wedge h_{i,k_i}$ in $head(r)$ will also be called a *head element*, and its probability p_i will be denoted by $P(h_{i,1} \wedge \ldots \wedge h_{i,k_i} \mid r)$. The meaning of a rule is that whenever $b_1\theta, \ldots, b_m\theta$ holds for a substitution θ in the current state I_t, exactly one head element $h_{i,1}\theta \wedge \ldots \wedge h_{i,k_i}\theta$ is chosen from $head(r)$ and all its conjuncts $h_{i,j}\theta$ are added to the next state I_{t+1}.

Example 1. Consider the following CPT-theory for the *travian world* domain:

$$conq(P, C) : 0.039 \vee nil : 0.961 \longleftarrow conq(P, C'), city(C', \_, \_, \_, P'), city(C, \_, \_, \_, P')$$

The rule encodes that a player is likely to conquer a city of a player he or she already attacked in the previous time step.

2 Experimental Evaluation[1]

Massively Multiplayer Online Game Domain As a second evaluation domain, we consider the large-scale massively multiplayer online strategy game Travian[2]. Game worlds feature thousands of players, game artifacts such as cities, armies, and resources, and social player interaction in alliances. Game states in Travian are complex and richly structured, and transitions between game states highly stochastic as they are determined by player actions. We have logged the state of a "live" game server over several months, recording high-level game states as visualized in Figure 2 (2-4). We address different learning tasks in the Travian domain, such as predicting player actions (prediction setting) and identifying groups of cooperating alliances (classification setting).

2.1 Experiments in the Massively Multiplayer Online Game Domain

In Travian, players are spread over several independent *game worlds*, with approximately 20.000–30.000 players interacting in a single world. Travian gameplay follows

[1] The implementation, models and data will be made available soon at http://www.ingothon.de/
[2] www.travian.com;www.traviangames.com

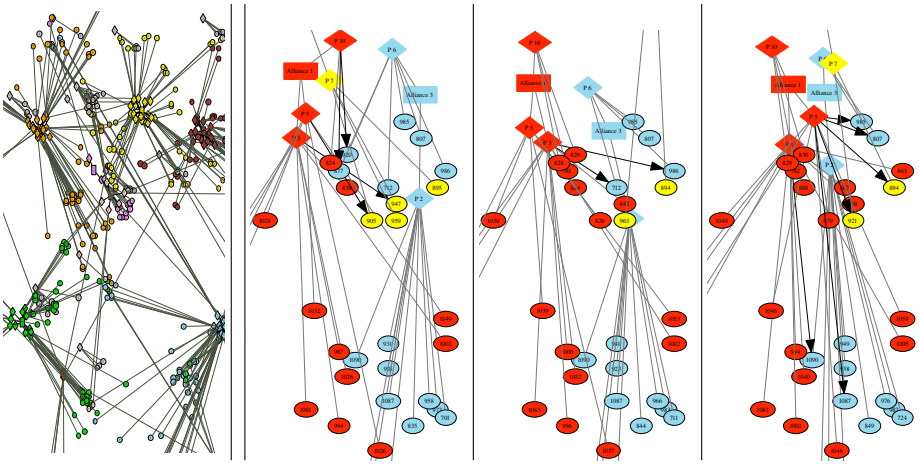

Fig. 2: (Left) High-level view of a (partial) game world in Travian.
(Other) Travian game dynamics visualized as changes in the game graph (for $t = 1, 2, 3$). Bold arrows indicate conquest attacks by a player on a particular city.

a classical strategy game setup on a large *grid-map*. Each player starts with a single *city* . During the course of the game, players harvest *resources* from the environment, construct *buildings*, research *technologies*, or found new cities on free tiles of the map. Additionally, players can build different military.

In the following, we will take a more high-level view of the game and focus on modeling player interaction and cooperation in alliances rather than low-level game elements such as resources, troops and buildings. Alliances constitute social networks, where diplomacy is used to settle conflicts and players compete for an influential role. Figure 2 shows a (partial) high-level view, represented as a graph structure relating cities, players and alliances which we will refer to as a *game graph*.

The dynamics of the game are illustrated in Figure 2 (2-4). Here, three players from the red alliance launch an attack against territory currently held by the blue and yellow alliances.

Data Collection and Preprocessing The data was collected from a "live" Travian server with approximately 25.000 active players. Over a period of three months (December 2007 till February 2008), high-level data about the current state was collected once every 24 hours. The data was represented using predicates $city(City, X, Y, Size, Player)$, $allied(Player, Alliance)$, $conq(Player, City)$ and $alliance\_change(Player, Alliance)$. The background knowledge contained the discretized distance $distance(C_1, C_2, D)$ with $D \in \{near, medium, far\}$ between cities.

Classification Experiments As a classification setting, we consider the problem of identifying so-called *meta-alliances* in Travian, introduced by Karwath et al. [2]. A meta-alliance is a group of alliances that closely cooperate. We manually identified meta-alliances in the collected game data based on the alliance names for instance the alliances *'.˜A˜.'*, *'.=A=.'*, and *'.-A-.'* are different wings of the same meta-alliance.

From all available game data 30 sequences were extracted constituting positive examples. A further 60 negative examples were obtained by giving the wrong meta-alliance information.

We hand-coded a simple CPT-theory that encodes a few basic features that player of the same meta-alliances behave like alliance partners, for example

$$conq(C, P1) : 0.0061 \vee nil : 0.9939 \quad \longleftarrow city(C, \_, \_, P2), player(P2, \_, A1), \\ player(P1, \_, A2), \neg alliance\_partners(A1, A2).$$

states that the player $P1$ attacks a city C of a player $P2$ who is not his alliance partner.

A CPT-theory can be used for classification given positive and negative training sequences by learning the parameters of two CPT-theories $\mathcal{T}_+$ and $\mathcal{T}_-$. The likelihood of a test sequence S under the positive and negative models, $P(S \mid \mathcal{T}_+)$ and $P(S \mid \mathcal{T}_-)$ is evaluated, and the class for which this likelihood is higher is predicted.

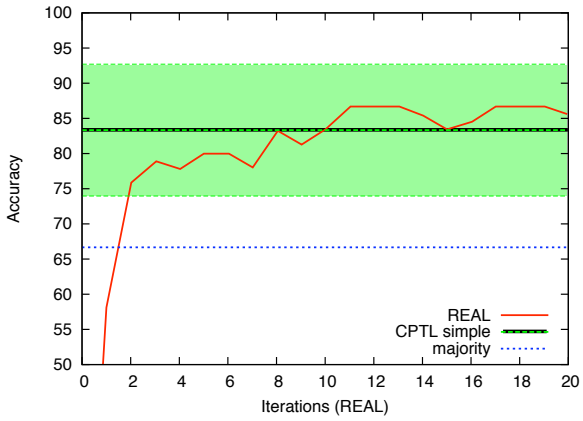

Fig. 3: Classification accuracy for the BOOSTEDREAL system (see [2]) and CPT-L for the meta-alliance problem in the massively multiplayer online game domain. For BOOSTEDREAL, accuracy is a function of the boosting iteration (shown on the x-axis). For CPT-L, standard deviation over the cross-validation folds is indicated by the green-shaded area. Classification accuracy of the majority-class predictor is also shown.

Figure 3 compares the results obtained for CPT-L with those of the BOOSTEDREAL system [2]. BOOSTEDREAL is a state-of-the-art system for classification of (relational)

sequences by alignment, which uses a discriminative approach based on boosting the reward model used in the alignment algorithm [2]. Even so BOOSTEDREAL is a discriminative and much more complicated approach tailored toward classification problems, it does not perform significantly better.Furthermore are the resulting models harder to interpret, as the boosted reward function is represented as an ensemble of relational regression trees. Figure 3 shows that CPT-L, at 82.22% with standard deviation of 9.37, achieves a slightly but not significantly lower accuracy than the best observed result for BOOSTEDREAL. Overall, we can conclude from this experiment that even with the simple rule set used, CPT-L is able to learn a model that captures useful information about the positive and negative class, and achieves similar accuracies as other state-of-the-art sequence classification schemes. Learning a single model in this domain takes under 2 minutes.

References

1. Vennekens, J., Denecker, M., Bruynooghe, M.: Representing Causal Information About a Probabilistic Process. In: Logics In Artificial Intelligence. Volume 4160 of Lecture Notes in Computer Science. (2006) 452–464
2. Karwath, A., Kersting, K., Landwehr, N.: Boosting relational sequence alignments. In: Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008). (2008)