Entanglement – A Measure for the Complexity of Directed Graphs With Applications to Logic and Games[⋆]

Dietmar Berwanger and Erich Grädel

Mathematische Grundlagen der Informatik, RWTH Aachen

Abstract. We propose a new parameter for the complexity of finite directed graphs which measures to what extent the cycles of the graph are intertwined. This measure, called entanglement, is defined by way of a game that is somewhat similar in spirit to the robber and cops games used to describe tree width, directed tree width, and hypertree width. Nevertheless, on many classes of graphs, there are significant differences between entanglement and the various incarnations of tree width.
Entanglement is intimately connected to the computational and descriptive complexity of the modal μ-calculus. On the one hand, the number of fixed point variables needed to describe a finite graph up to bisimulation is captured by its entanglement. This plays a crucial role in the proof that the variable hierarchy of the μ-calculus is strict.
In addition to this, we prove that parity games of bounded entanglement can be solved in polynomial time. Specifically, we establish that the complexity of solving a parity game can be parametrised in terms of the minimal entanglement of a subgame induced by a winning strategy.

1 Entanglement: How to catch a thief

Let $\mathcal{G} = (V, E)$ be a finite directed graph. The *entanglement* of $\mathcal{G}$, denoted $\mathrm{ent}(\mathcal{G})$, measures to what extent the cycles of $\mathcal{G}$ are entangled. We define the entanglement by way of a game, played by a thief against k detectives on $\mathcal{G}$ according to the following rules. Initially the thief selects an arbitrary position v_0 of $\mathcal{G}$ and the detectives are outside the graph. In any move the detectives may either stay where they are, or place one of them on the current position v of the thief. The thief, in turn, has to move to a successor $w \in vE$ that is not occupied by a detective. If no such position exists, the thief is caught and the detectives have won. Note that the thief sees the move of the detectives before she decides on her own move, and that she has to leave her current position no matter whether the detectives stay where they are or not. The entanglement of $\mathcal{G}$ is the minimal number $k \in \mathbb{N}$ such that k detectives have a strategy to catch the thief on $\mathcal{G}$.

⋆ This research has been partially supported by the European Community Research Training Network "Games and Automata for Synthesis and Validation" (GAMES)

The entanglement is an interesting measure on *directed* graphs. To deal with undirected graphs, we view undirected edges $\{u,v\}$ as pairs (u,v) and (v,u) of directed edges. In the following a graph is always meant to be directed.

To get a feeling for this measure we collect a few simple observations concerning the entanglement of certain familiar graphs. The proofs are simple and left to the reader.

Proposition 1. *Let $\mathcal{G}$ be any finite directed graph.*

(1) $\mathrm{ent}(\mathcal{G}) = 0$ *if, and only if, $\mathcal{G}$ is acyclic.*
(2) *If $\mathcal{G}$ is the graph of a unary function, then* $\mathrm{ent}(\mathcal{G}) = 1$.
(3) *If $\mathcal{G}$ an undirected tree, then* $\mathrm{ent}(\mathcal{G}) \le 2$.
(4) *If $\mathcal{G}$ is the fully connected directed graph with n nodes, then* $\mathrm{ent}(\mathcal{G}) = n$.

Let C_n denote the directed cycle with n nodes. Given two graphs $\mathcal{G} = (V,E)$ and $\mathcal{G}' = (V',E')$ their asynchronous product is the graph $\mathcal{G} \times \mathcal{G}' = (V \times V', F)$ where $F = \{(uu', vv') : [(u,v) \in E \wedge u' = v'] \vee [u = v \wedge (u'v') \in E']\}$.

Note, that $T_{mn} := C_m \times C_n$ is the $(m \times n)$-torus or, to put it differently, the graph obtained from the directed $(m+1) \times (n+1)$-grid by identifying the left and right border and the upper and lower border.

Proposition 2. (1) *For every n,* $\mathrm{ent}(T_{nn}) = n$.
(2) *For every $m \neq n$,* $\mathrm{ent}(T_{mn}) = \min(m,n) + 1$.

Proof. On T_{nn}, n detectives can catch the thief by placing themselves on a diagonal, thus blocking every row and every column of the torus. On the other side, it is obvious that the thief can escape against $n-1$ detectives.

On T_{mn} with $m < n$, m detectives are needed to block every row, and an additional detective forces the thief to leave any row after at most n moves, so that she finally must run into a detective. Again, it is obvious that the thief escapes if there are less than $m+1$ detectives. □

The following proposition characterises the graphs with entanglement one.

Proposition 3. *The entanglement of a directed graph is one, if and only if, the graph is not acyclic, and in every strongly connected component, there is a node whose removal makes the component acyclic.*

Proof. On any graph with this property, one detective catches the thief by placing himself on the critical node in the current strongly connected component when the thief passes there. The thief will have to return to this node or leave the current component. Eventually she will be caught in a terminal component.

Conversely if there is a strongly connected component without such a critical node, then the thief may always proceed from her current position towards an unguarded cycle and thus escape forever. □

Corollary 4. *For $k = 0$ and $k = 1$, the problem whether a given graph has entanglement k is* NLOGSPACE*-complete.*

To compute upper bounds on the entanglement of certain interesting graphs, we can use the following sufficient criterion for the existence of a winning strategy for k detectives. For any $k \in \mathbb{N}$, let $[k] := \{0, \dots, k-1\}$.

Lemma 5. *Let $\mathcal{G} = (V, E)$ be a finite directed graph such that, for some $k \in \mathbb{N}$, there exists a partial labelling $i : V \to [k]$ under which every strongly connected subgraph $\mathcal{C} \subseteq \mathcal{G}$ contains a vertex v whose label is unique in C, that is, $i(v) \neq i(w)$ for all $w \in \mathcal{C}$. Then* $\mathrm{ent}(\mathcal{G}) \leq k$.

Proof. We may interpret the labelling i as a memoryless strategy for the detectives, indicating at every position v occurring in a play, that detective $i(v)$ shall be posted there, or that no detective shall move if $i(v)$ is undefined. Towards a contradiction, suppose that although the detectives move according to strategy i, the thief can escape, that is, she succeeds to form an infinite path without meeting any detective. Let $\mathcal{C}$ be the set of positions visited infinitely often by this path. Clearly, $\mathcal{C}$ induces in $\mathcal{G}$ a strongly connected subgraph. Let $v \in \mathcal{C}$ be a node whose label $i(v)$ is unique in $\mathcal{C}$. According to the strategy described by i, detective $i(v)$ remains at v once the play has stabilised in $\mathcal{C}$. But since the thief visits every position in $\mathcal{C}$ infinitely often, she is caught at v. □

Proposition 6. *For every n, the undirected $(n \times n)$-grid has entanglement at most $3n$.*

Proof. Consider the labelling $i : [n] \times [n] \to [3n]$ obtained by first assigning the values $0, \dots, n$ to the horizontal median of the grid, i.e., $i(\lfloor \frac{n}{2} \rfloor, j) := j$ for all $j \in [n]$. For the two $\frac{n}{2} \times n$ grids obtained when removing the positions already labelled, we proceed independently and assign the values $n, \dots, n + \frac{n}{2}$ to their vertical medians, and so on, in step k applying the procedure to the still unlabelled domain consisting of 2^k many $\frac{n}{2^k} \times \frac{n}{2^k}$ disconnected grids. It is easy to verify that the labelling obtained this way satisfies the criterion of Lemma 5. □

2 Entanglement versus tree width

The definition of entanglement is reminiscent of robber and cops games introduced by Seymour and Thomas in [10] for characterising tree width, and Johnson, Robertson, Seymour, and Thomas [6] for directed tree width. However, entanglement is a quite different, and for some purposes more accurate, measure than tree width and directed tree width.

This becomes apparent on trees with back-edges which also play an important role in our analysis of the variable hierarchy of the modal μ-calculus. It is easy to see that the directed tree width of any tree with back-edges is one. However, we will see that the entanglement of trees with back-edges can be arbitrarily large.

We now discuss the relationship between (undirected) tree width and entanglement. First, we observe that acyclic graphs (that have entanglement 0) can of course have arbitrary tree width. On the other hand we prove that the entanglement of a graph can be bounded by its tree width times the logarithm of its size.

Proposition 7. *For any finite undirected graph $\mathcal{G}$ of tree width k, we have that* $\mathrm{ent}(\mathcal{G}) \leq (k+1) \cdot \log |G|$.

Proof. By definition, every graph $\mathcal{G} = (V, E)$ of tree width k can be decomposed as a tree $\mathcal{T}$ labelled with subsets of at most $k+1$ elements of V, called *blocks*, such that (1) every edge $\{u, v\} \in E$ is included in some block and (2) for any element $v \in V$ the set of blocks containing v is connected.

In every subtree $\mathcal{S}$ of such a decomposition tree, there exists a node s, we may call it the *centre* of $\mathcal{S}$, which balances $\mathcal{S}$ in the sense that the subtree rooted at s and its complement carry almost the same number of vertices (differences up to k are admissible). Consider now the following memoryless detective strategy. First, all vertices in the centre s of the decomposition tree receive indices $0, \dots, k$. Then, we repeat the process independently for the two subtrees (i.e., the one rooted in s and its complement) and assign to the vertices in their respective centres indices from $k+1, \dots, 2k+2$. The process ends when all vertices of $\mathcal{G}$ are labelled. In this way, at most $(k+1) \log |V|$ detective indices are assigned. Since the blocks of a tree decomposition separate the graph, every strongly connected subgraph of $\mathcal{G}$ will contain at least one unique label. This shows that the constructed labelling indeed represents a memoryless strategy for at most $(k+1) \log |V|$ detectives. □

However, bounded tree width does not imply bounded entanglement.

Proposition 8. *There exist graphs with tree width two that have arbitrarily large entanglement.*

Proof. Let $\mathcal{T}_k^{\downarrow}$ be the full binary tree of depth k with edges oriented downwards, and let $\mathcal{T}_k^{\uparrow}$ be the same tree with edges oriented upwards. Every node $v^{\downarrow} \in \mathcal{T}_k^{\downarrow}$ has a *double* $v^{\uparrow} \in \mathcal{T}_k^{\uparrow}$, and vice versa. The graph $G(2, k)$ is constructed by taking the union $\mathcal{T}_k^{\downarrow} \cup \mathcal{T}_k^{\uparrow}$, adding edges from each leaf to its double (in both directions), and adding the edges $(u^{\uparrow}, v^{\downarrow})$ for each edge $(u^{\uparrow}, v^{\uparrow})$ of $\mathcal{T}_k^{\uparrow}$. It is easy to see that $G(2, k)$ has tree width 2.

We claim that $\mathrm{ent}(G(2, k)) > k$. To prove this we describe a strategy by which the thief escapes against k detectives. We call a path in $G(2, k)$ *free* if all nodes on the path and all their doubles are unguarded by the detectives. We say that a node is *blocked* if both the node and its double are guarded. The thief moves according to the following strategy: *at a leaf $w^{\uparrow}$, she selects an ancestor $u^{\downarrow}$ of $w^{\downarrow}$ from which there is a free path to a leaf $v^{\downarrow}$. She goes to $v^{\downarrow}$ by moving upwards through $\mathcal{T}_k^{\uparrow}$, stepping over to $u^{\downarrow}$ and moving downwards through $\mathcal{T}_k^{\downarrow}$. Finally she steps over to $v^{\uparrow}$.*

With this strategy, the thief is never below a blocked node. A leaf has (including itself) $k+1$ ancestors in $\mathcal{T}_k^{\downarrow}$, so there is always an ancestor with a free path to a leaf. Thus, the thief can maintain this strategy and escape forever. □

3 Trees with back-edges and partial unravellings

Let $\mathcal{T} = (V, E)$ be a directed tree. We write $\preceq_E$ for the associated partial order on $\mathcal{T}$. Note that $\preceq_E$ is just the reflexive transitive closure of E.

Definition 9. A directed graph $\mathcal{T} = (V, F)$ is a *tree with back-edges* if there is a partition $F = E \cup B$ of the edges into tree-edges and back-edges such that (V, E) is indeed a directed tree, and whenever $(u, v) \in B$, then $v \preceq_E u$.

The following observation shows that, up to the choice of the root, the decomposition into tree-edges and back-edges is unique.

Lemma 10. *Let $\mathcal{T} = (V, F)$ be a tree with back-edges and $v \in V$. Then there exists at most one decomposition $F = E \cup B$ into tree-edges and back-edges such that (V, E) is a tree with root v.*

Definition 11. Let $\mathcal{T} = (V, F)$ be a tree with back-edges, with decomposition $F = E \cup B$ into tree-edges and back-edges. The *feedback* of a node v of $\mathcal{T}$ is the number of ancestors of v that are reachable by a back-edge from a descendant of v. The feedback of $\mathcal{T}$, denoted $\text{fb}(T)$ is the maximal feedback of nodes on $\mathcal{G}$. More formally,

$$\text{fb}(T) = \max_{v \in V} |\{u \in V : \exists w (u \preceq_E v \preceq_E w \wedge (w, u) \in B)\}|.$$

We call a back edge (w, u), and likewise its target u, *active* at a node v in $\mathcal{T}$, if $u \preceq_E v \preceq_E w$.

Note that the feedback of $\mathcal{T}$ may depend on how the edges are decomposed into tree-edges and back-edges, i.e. on the choice of the root. Consider, for instance the following graph C_3^+ (the cycle C_3 with an additional self-loop on one of its nodes). Clearly, for every choice of the root, C_3^+ is a tree with two back-edges. If the node with the self-loop is taken as the root, then the feedback is 1, otherwise it is 2.

Lemma 12. *Let $\mathcal{T} = (V, E, B)$ be a tree with back-edges of feedback k. Then there exists a partial labelling $i : V \mapsto \{0, \ldots, k-1\}$ assigning to every target u of a back edge an index $i(u)$ in such a way that no two nodes u, u' that are active at the same node v have the same index.*

Proof. The values of this labelling are set while traversing the tree in breadth-first order. Notice that every node u with an incoming back-edge is active at itself. As $\mathcal{T}$ has feedback k, there can be at most $k-1$ other nodes active at u. All of these are ancestors of u, hence their index is already defined. There is at least one index which we can assign to u so that no conflict with the other currently active nodes arises.

Lemma 13. *The entanglement of a tree with back-edges is at most its feedback:* $\text{ent}(\mathcal{T}) \leq \text{fb}(\mathcal{T})$.

Proof. Suppose that $\text{fb}(\mathcal{T}) = k$. By Lemma 12 there is a labelling i of the targets of the back-edges in $\mathcal{T}$ by numbers $0, \ldots, k-1$ assigning different values to any two nodes u, u' that are active at the same node v. This labelling induces the following strategy for the k detectives: at every node v reached by the thief, send detective number $i(v)$ to that position or, if the value is undefined, do nothing. By induction over the stages of the play, we can now show that this strategy maintains the following invariant: at every node v occurring in a play on $\mathcal{T}$, all active nodes $u \neq v$ are occupied and, if the current node is itself active, a detective is on the way. To see this, let us trace the evolution of the set $Z \subseteq T$ of nodes occupied by a detective. In the beginning of the play, Z is empty. A node v can be included into Z if it is visited by the thief and active with regard to itself. At this point, our strategy appoints detective $i(v)$ to move to v. Since, by construction of the labelling, the designated detective $i(v)$ must come from a currently inactive position and, hence, all currently active positions except v remain in Z. But if every node which becomes active is added to Z and no active node is ever given up, the thief can never move along a back edge, so that after a finite number of steps she reaches a leaf of the tree and loses. But this means that we have a winning strategy for k detectives, hence $\mathrm{ent}(\mathcal{T}) \leq k$. $\square$

Note however, that the entanglement of a tree with back-edges can be much smaller than its feedback. A simple example are paths with back-edges: let $P_n = (\{0, \ldots, n-1\}, E_n, B_n)$ be the path with n nodes and all possible back-edges, i.e., $E_n = \{(i, i+1) : i < n-1\}$ and $B_n = \{(i,j) : i \geq j\}$. Obviously, $\mathrm{fb}(P_n) = n$, but two detectives suffice to catch the thief on P_n.

It is well-known that every graph $\mathcal{G}$ can be unravelled from any node v to a tree $\mathcal{T}_{G,v}$ whose nodes are the paths in $\mathcal{G}$ from v. Clearly $\mathcal{T}_{G,v}$ is infinite unless $\mathcal{G}$ is finite and no cycle in $\mathcal{G}$ is reachable from v. A *finite unravelling* of a (finite) graph $\mathcal{G}$ is defined in a similar way, but rather than an infinite tree, it produces a finite tree with back-edges. To construct a finite unravelling we proceed as in the usual unravelling process with the following modification: whenever we have a path $v_0 v_1 \ldots v_n$ in $\mathcal{G}$ with corresponding node $\overline{v} = v_0 v_1 \ldots v_n$ in the unravelling, and a successor w of v_n that coincides with v_i (for any $i \leq n$), then we may, instead of creating the new node $\overline{v}w$ (with a tree-edge from $\overline{v}$ to $\overline{v}w$) put a back-edge from $\overline{v}$ to its ancestor $v_0 \ldots v_i$. Clearly this process is nondeterministic. In this way, any finite graph can be unravelled, in many different ways, to a finite tree with back-edges.

Observe that different finite unravellings of a graph may have different feedback and different entanglement. Clearly the entanglement of a graph is bounded by the entanglement of its finite unravellings. Indeed a winning strategy for k detectives on a finite unravelling of $\mathcal{G}$ immediately translates to a winning strategy on $\mathcal{G}$.

Proposition 14. *The entanglement of a graph is the minimal feedback (and the minimal entanglement) of its finite unravellings:*

$$\begin{aligned} \mathrm{ent}(\mathcal{G}) &= \min\{\mathrm{fb}(\mathcal{T}) : \mathcal{T} \textit{ is a finite unravelling of } \mathcal{G}\} \\ &= \min\{\mathrm{ent}(\mathcal{T}) : \mathcal{T} \textit{ is a finite unravelling of } \mathcal{G}\}. \end{aligned}$$

Proof. For any finite unravelling $\mathcal{T}$ of a graph $\mathcal{G}$, we have $\mathrm{ent}(\mathcal{G}) \leq \mathrm{ent}(\mathcal{T}) \leq \mathrm{fb}(\mathcal{T})$. It remains to show that for any graph $\mathcal{G}$ there exists some finite unravelling $\mathcal{T}$ with $\mathrm{fb}(\mathcal{T}) \leq \mathrm{ent}(\mathcal{G})$.

To prove this, we view winning strategies for the detectives as descriptions of finite unravellings. A strategy for k detectives tells us, for any finite path πv of the thief whether a detective should be posted at the current node v, and if so, which one. Such a strategy can be represented by a partial function g mapping finite paths in $\mathcal{G}$ to $\{0, \ldots, k-1\}$. On the other hand, during the process of unravelling a graph to a (finite) tree with back edges, we need to decide, for every successor v of the current node, whether to create a new copy of v or to return to a previously visited one, if any is available. To put this notion on a formal ground, we define an *unravelling function* for a rooted graph $\mathcal{G}, v_0$ as a partial function ρ between finite paths from v_0 through $\mathcal{G}$, mapping any path $v_0, \ldots, v_{r-1}, v_r$ in its domain to a strict prefix $v_0, v_1, \cdots, v_{j-1}$ such that $v_{j-1} = v_r$. Such a function gives rise to an unravelling of $\mathcal{G}$ in the following way: we start at the root and follow finite paths through $\mathcal{G}$. Whenever the current path π can be prolonged by a position v and the value of ρ at πv is undefined, a fresh copy of v corresponding to πw is created as a successor of π. In particular, this always happens if v was not yet visited. Otherwise, if $\rho(\pi\, v)$ is defined, then the current path π is bent back to its prefix $\rho(\pi)$ which also corresponds to a copy of v. Formally, the unravelling of $\mathcal{G}$ *driven* by ρ is the tree with back edges $\mathcal{T}$ defined as follows:

– the domain of $\mathcal{T}$ is the smallest set T which contains v_0 and for each path $\pi \in T$, it also contains all prolongations πv in $\mathcal{G}$ at which ρ is undefined;
– the tree-edge partition is

$$E^{\mathcal{T}} := \{\, (v_0, \ldots, v_{r-1},\ v_0, \ldots, v_{r-1}, v_r) \in T \times T \mid (v_{r-1}, v_r) \in E^{\mathcal{G}} \,\};$$

– for all paths $\pi := v_0, \ldots, v_{r-1} \in T$ where $\rho(\pi v)$ is defined, the back-relation $B^{\mathcal{T}}$ contains the pair $(\pi,\ \rho(\pi v))$ if $(v_{r-1}, v) \in E^{\mathcal{G}}$.

We are now ready to prove that every winning strategy g for the k detectives on $\mathcal{G}, v_0$ corresponds to an unravelling function ρ for $\mathcal{G}, v_0$ that controls a finite unravelling with feedback k.

Note that the strategy g gives rise to a k-tuple $(g_0, \ldots, g_{k-1})$ of functions mapping every initial segment π of a possible play according to g to a k-tuple $(\, g_0(\pi), \ldots, g_{k-1}(\pi)\,)$ where each $g_i(\pi)$ is a prefix of π recording the state of the play (i.e., the current path of the thief) at the last move of detective i.

Now, for every path π and possible prolongation by v, we check whether, after playing π, there is any detective posted at v. If this is the case, i.e, when, for some i, the end node of $g_i(\pi)$ is v, we set $\rho(\pi\, v) := \pi_i$. Otherwise we leave the value of ρ undefined at π, v. It is not hard to check that, if g is a winning strategy for the detectives, the associated unravelling is finite and has feedback k. $\square$

4 Descriptive complexity

The modal μ-calculus L_μ introduced by Kozen [8] is a highly expressive formalism which extends basic modal logic with monadic variables and binds them to extremal fixed points of definable operators.

Syntax. For a set $\textsc{act}$ of actions, a set $\textsc{prop}$ of atomic propositions, and a set $\textsc{var}$ of monadic variables, the formulae of L_μ are defined by the grammar

$$\varphi ::= \text{false} \mid \text{true} \mid p \mid \neg p \mid X \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \langle a \rangle \varphi \mid [a]\varphi \mid \mu X.\varphi \mid \nu X.\varphi$$

where $p \in \textsc{prop}$, $a \in \textsc{act}$, and $X \in \textsc{var}$. An L_μ-formula in which no universal modality $[a]\varphi$ occurs is called *existential.*

The number of variables occurring in a formula provides a relevant measure of its conceptual complexity. For any $k \in \mathbb{N}$, the *k-variable fragment* $\mathrm{L}_\mu[k]$ of the μ-calculus is the set of formulae $\psi \in \mathrm{L}_\mu$ that contain at most k distinct variables.

Semantics. Formulae of L_μ are interpreted on transition systems, or Kripke structures. Formally, a transition system $\mathcal{K} = \big(V, (E_a)_{a\in\textsc{act}}, (V_p)_{p\in\textsc{prop}}\big)$ is a coloured graph with edges labelled by action and vertices labelled by atomic propositions. Given a sentence ψ and a structure $\mathcal{K}$ with state v, we write $\mathcal{K}, v \models \psi$ to denote that ψ holds in $\mathcal{K}$ at state v. The set of states $v \in K$ such that $\mathcal{K}, v \models \psi$ is denoted by $[\![\psi]\!]^{\mathcal{K}}$.

Here, we only define $[\![\psi]\!]^{\mathcal{K}}$ for fixed-point formulae ψ. Towards this, note that a formula $\psi(X)$ with a monadic variable X defines on every transition structure $\mathcal{K}$ (providing interpretations for all free variables other than X occurring in ψ) an operator $\psi^{\mathcal{K}} : \mathcal{P}(K) \to \mathcal{P}(K)$ assigning to every set $X \subseteq K$ the set $\psi^{\mathcal{K}}(X) := [\![\psi]\!]^{\mathcal{K},X} = \{v \in K : (\mathcal{K}, X), v \models \psi\}$. As X occurs only positively in ψ, the operator $\psi^{\mathcal{K}}$ is *monotone* for every $\mathcal{K}$, and therefore, by a well-known theorem due to Knaster and Tarski, has a least fixed point $\mathrm{lfp}(\psi^{\mathcal{K}})$ and a greatest fixed point $\mathrm{gfp}(\psi^{\mathcal{K}})$. Now we put

$$[\![\mu X.\psi]\!]^{\mathcal{K}} := \mathrm{lfp}(\psi^{\mathcal{K}}) \text{ and } [\![\nu X.\psi]\!]^{\mathcal{K}} := \mathrm{gfp}(\psi^{\mathcal{K}}).$$

As a modal logic, the μ-calculus distinguishes between transitions structures only up to behavioural equivalence, captured by the notion of bisimulation.

Definition 15. A *bisimulation* between two transition structures $\mathcal{K}$ and $\mathcal{K}'$ is a simulation Z from $\mathcal{K}$ to $\mathcal{K}'$ so that the inverse relation Z^{-1} is a simulation from $\mathcal{K}'$ to $\mathcal{K}$. Two transition structures $\mathcal{K}, u$ and $\mathcal{K}', u'$ are *bisimilar*, denoted $\mathcal{K}, u \sim \mathcal{K}', u'$, if there is a bisimulation Z between them, with $(u, u') \in Z$.

An important model-theoretic feature of modal logics is the *tree model property* meaning that every satisfiable formula is satisfiable in a tree. This is a straightforward consequence of bisimulation invariance, since $\mathcal{K}, u$ is bisimilar to its *infinite unravelling*, i.e., a tree whose nodes correspond to the finite paths in $\mathcal{K}, u$. Every such path π inherits the atomic propositions of its last node v; for every node w reachable from v in $\mathcal{K}$ via an a transition, π is connected to its prolongation by w via an a-transition. Notice that in terms of our notion of unravelling defined in the proof of Proposition 14, the infinite unravelling of a system is just the unravelling driven by a function defined nowhere.

The entanglement of a transition system $\mathcal{K} = \big(V, (E_a)_{a\in\textsc{act}}, (V_p)_{p\in\textsc{prop}}\big)$ is the entanglement of the underlying graph (V, E) where $E = \bigcup_{a\in\textsc{act}} E_a$. We now show that every transition structure of entanglement k can be described, up to bisimulation, in the μ-calculus using only k fixed-point variables.

Proposition 16. *Let $\mathcal{K}$ be a finite transition system with* $\operatorname{ent}(\mathcal{K}) = k$*. For any node v of $\mathcal{K}$, there is a formula $\psi_v \in L_\mu[k]$ such that*

$$\mathcal{K}', v' \models \psi_v \;\Leftrightarrow\; \mathcal{K}', v' \sim \mathcal{K}, v.$$

Proof. According to Proposition 14, the system $\mathcal{K}$ can be unravelled from any node v_0 to a finite tree $\mathcal{T}$ with back-edges, with root v_0 and feedback k. Clearly $\mathcal{T}, v_0 \sim \mathcal{K}, v_0$. Hence, it is sufficient to prove the proposition for $\mathcal{T}, v_0$. For every action $a \in \text{ACT}$, the transitions in $\mathcal{T}$ are partitioned into tree-edges and back-edges $E_a \uplus B_a$.

Let $i : \mathcal{T} \mapsto \{0, \dots, k-1\}$ be the partial labelling of $\mathcal{T}$ defined in Lemma 12. At hand with this labelling, we construct a sequence of formulae $(\psi_v)_{v \in T}$ over fixed-point variables $X_0, \dots, X_{k-1}$ while traversing the nodes of $\mathcal{T}$ in reverse breadth-first order.

The atomic type of any node v is described by the formula

$$\alpha_v := \bigwedge_{\substack{p \in \text{PROP} \\ v \in V_p}} p \;\wedge\; \bigwedge_{\substack{p \in \text{PROP} \\ v \notin V_p}} \neg p.$$

To describe the relationship of v with its successors, let

$$\varphi_v := \alpha_v \wedge \bigwedge_{a \in \text{ACT}} \Bigg(\bigwedge_{(v,w) \in E_a} \langle a \rangle \psi_w \wedge \bigwedge_{(v,w) \in B_a} \langle a \rangle X_{i(w)} \\ \wedge\; [a] \Big(\bigvee_{(v,w) \in E_a} \psi_w \vee \bigvee_{(v,w) \in B_a} X_{i(w)} \Big) \Bigg).$$

If v has an incoming back-edge, we set $\psi_v := \nu X_{i(v)} \,.\, \varphi_v$, if this is not the case we set $\psi_v := \varphi_v$. Note that since we proceed from the leaves of $\mathcal{T}$ to the root, this process is well-defined, and that in ψ_v the variables $X_{i(u)}$ occur free, for any node $u \neq v$ that is active at v. In particular the formula ψ_{v_0}, corresponding to the root of $\mathcal{T}$, is closed.

It remains to prove that $\mathcal{K}', v' \models \psi_{v_0} \;\Leftrightarrow\; \mathcal{K}', v' \sim \mathcal{T}, v_0$. We first show that $\mathcal{T}, v_0 \models \psi_{v_0}$, and hence $\mathcal{K}', v' \models \psi_{v_0}$ for any $\mathcal{K}', v' \sim \mathcal{T}, v_0$. To see this we prove that Verifier has a winning strategy for the associated model checking game.

Note that, since ψ_{v_0} has only greatest fixed points, any infinite play of the model checking game is won by Verifier. It thus suffices to show that from any position of form (v, φ_v), Verifier has a strategy to make sure that the play proceeds to a next position of form (w, φ_w), unless Falsifier moves to position (v, α_v) and then loses in the next move. But by the construction of the formula, it is obvious that Verifier can play so that any position at which she has to move has one of the following three types:

(1) $(v, \langle a \rangle \psi_w)$, where $(v, w) \in E_a$. In this case, Verifier moves to position (w, ψ_w).
(2) $(v, \langle a \rangle X_{i(w)})$, where $(v, w) \in B_a$. In this case Verifier moves to $(w, X_{i(w)})$.
(3) $(w, \bigvee_{(v,w) \in E_a} \psi_w \vee \bigvee_{(v,w) \in B_a} X_{i(w)})$ where $w \in vE_a \cup vB_a$. In this case, Verifier selects the appropriate disjunct and moves to either (w, ψ_w) or $(w, X_{i(w)})$.

In all cases the play will proceed to (w, φ_w). Hence, Falsifier can force a play to be finite only by moving to a position (v, α_v). Otherwise the resulting play is infinite and thus also won by Verifier.

For the converse, suppose that $\mathcal{K}', v' \not\sim \mathcal{T}, v_0$. Since $\mathcal{T}$ is finite, the non-bisimilarity it witnessed by a finite stage. That is, there is a basic modal formula separating $\mathcal{K}', v'$ from $\mathcal{T}, v_0$, and Falsifier can force the model checking game for ψ_{v_0} on $\mathcal{K}', v'$ in finitely many moves to a position of form (w', α_w) such that w and w' have distinct atomic types. This proves that $\mathcal{K}', v' \not\models \psi_{v_0}$. □

As the entanglement of a transition system regards only the underlying graph, one can easily find examples of high entanglement that can be described with very few variables. For instance, in a transition structure over a strongly connected finite graph with no atomic propositions and only a single action a, all states are bisimilar, and can be described by $\nu X.(\langle a \rangle X \wedge [a]X)$, regardless of the entanglement of the underlying graph. Nevertheless, the following theorem establishes a strong relationship between the notion of entanglement and the descriptive complexity of L_μ.

Theorem 17 ([2])**.** *Every strongly connected graph of entanglement k can be labelled in such a way that no μ-calculus formula with less than k variables can describe the resulting transition structure, up to simulation.*

This theorem, which generalises a result of [3], provides the witnesses for the expressive strictness of the μ-calculus variable hierarchy proved in [4].

5 Computational complexity

An intriguing open problem related to the μ-calculus regards the computational complexity of its evaluation problem: Given a formula ψ and a finite transition structure $\mathcal{K}, v$, decide whether ψ holds in $\mathcal{K}, v$. Equivalently, this problem can be phrased in terms of *parity games*, the natural evaluation games for L_μ [11].

Parity games are path-forming games played between two players on labelled graphs $\mathcal{G} = (V, V_0, E, \Omega)$ equipped with a *priority* labelling $\Omega : V \to \mathbb{N}$. All plays start from a given initial node v_0. At every node $v \in V_0$, the first player, called Player 0, can move to a successor $w \in vE$; at positions $v \in V_1 := V \setminus V_0$, his opponent Player 1 moves. Once a player gets stuck, he loses. If the play goes on infinitely, the winner is determined by looking at the sequence $\Omega(v_0), \Omega(v_1), \ldots$ of priorities seen during the play. In case the least priority appearing infinitely often in this sequence is even, Player 0 wins the play, otherwise Player 1 wins.

A *memoryless strategy* for Player i in a parity game $\mathcal{G}$ is a function σ that indicates a successor $\sigma(v) \in vE$ for every position $v \in V_i$. A strategy for a player is *winning*, if he wins every play starting in which he moves according to this strategy. The Memoryless Determinacy Theorem of Emerson and Jutla states that parity games are always determined with memoryless strategies.

Theorem 18 (Memoryless Determinacy, [5])**.** *In any parity game, one of the players has a memoryless winning strategy.*

Any memoryless strategy σ induces a subgraph $\mathcal{G}_\sigma$ of the original game graph. If σ is a winning strategy for a player, he wins every play on $\mathcal{G}_\sigma$. Since these subgames are small objects and it can be checked efficiently whether a player wins every play on a given graph, the winner of a finite parity game can be determined in NP $\cap$ co-NP. In general, the best known deterministic algorithms to decide the winner of a parity game have running times that are polynomial with respect to the size of the game graph, but exponential with respect to the number of different priorities occurring in the game [7]. However, for game graphs of bounded tree width, Obdrzalek has showed in [9], that the problem can be solved in polynomial time with respect to the the size of the graph, independently of the number of priorities.

In the remainder of this paper we will show that the entanglement of a parity game graph is a pivotal parameter for its computational complexity. To maintain the relationship between games and algorithms conceptually close, we base our analysis on alternating machines (for a comprehensive introduction, see e.g. [1]).

5.1 Alternating cycle detection

Many algorithmic issues in graph theory are related to the problem of cycle detection, typically, to determine whether a given graph contains a cycle satisfying certain properties. When alternation comes into play, that is, when we consider paths formed interactively, the questions become particularly interesting but often rather complex, too. In this framework, we will study the entanglement of a graph as a measure of how much memory is needed to determine whether a path formed on-the-fly enters a cycle.

As a basis for later development, let us first consider a procedure for deciding whether k detectives are sufficient to capture the thief on a given graph. The following algorithm represents a straightforward implementation of the game as an alternating algorithm, where the role of the thief is played by the existential player while the detectives are controlled by the universal player.

```
procedure Entanglement($\mathcal{G}, v_0, k$)
input graph $\mathcal{G} = (V, E)$, initial position $v_0$, candidate $k \leq |V|$
// accept iff ent($\mathcal{G}, v_0$) $\leq k$
$v := v_0, (d_i)_{i \in [k]} := \bot$;                    // current position of thief and detectives
do
  existentially guess  $i \in [k] \cup \{\text{pass}\}$        // appoint detective $i$ or pass
  if $i \neq$ pass then  $d_i := v$                           // guard current node
  if $vE \setminus \{d_i : i \in [k]\} = \emptyset$ then  accept
  else  universally choose  $v \in vE$;
repeat
```

Since this algorithm requires space only to store the current positions of the thief and the k detectives, it runs in alternating space $O((k + 1) \log |V|)$ which corresponds to deterministic polynomial time.

Lemma 19. *The problem of deciding, for a fixed parameter k, whether a given graph $\mathcal{G}$ has* ent$(\mathcal{G}) \leq k$ *can be solved in polynomial time.*

Notice that if we regard k as part of the input, the algorithm gives an EXPTIME upper bound for deciding the entanglement of a graph.

5.2 Parity games

Similar to the thief and detective game, the dynamics of a parity game consists in forming a path through a graph. However, while in the former game the detectives can influence the forming process only indirectly, by obstructing ways of return, in a parity game both players determine directly how the path is prolonged in their turn. Besides this dynamic aspect, also the objectives of players are quite different at a first sight. While the detectives aim at turning the play back to a guarded position, each player of a parity game tries to achieve that the least priority seen infinitely often on the path is of a certain parity.

The key insight which brings the two games to a common ground is the Memoryless Determinacy Theorem for parity games: whichever player has a winning strategy in a given game $\mathcal{G} = (V, V_0, E, \Omega)$, also has a memoryless one. This means, that either player may commit, for each reachable position $v \in V$ which he controls, to precisely one successor $\sigma(v) \in vE$ and henceforth follow this commitment in every play of $\mathcal{G}$ without risking any chance to win. It follows that, whenever a play returns to a previously visited position v, the winner can be established by looking at the least priority seen since the first occurrence of v. Therefore can view parity games on finite game graphs as path forming games of finite duration where the objective is to reach a cycle with minimal priority of a certain parity.

We obtain an immediate method to determine the winner of a parity game by simulating the players' moves while maintaining the history of visited positions in order to detect whether a cycle has been reached. To store the full history, an implementation of this method requires space $O(|V| \log |V|)$ in the worst case; since the procedure uses alternation to simulate the single game moves, this situates us in ASPACE($O(|V| \log |V|)$), or DTIME($|V|^{O(|V|)}$).

What makes this approach highly impractical is its extensive representation of the play's history. In fact, the power of alternation is limited to the formation of the path, while the history is surveyed in a deterministic way. We can significantly improve this by interleaving thief and detective games with parity games in such a way that the formation of cycles in history is surveyed interactively.

Intuitively, we may think of a parity game as an affair between three agents, Player 0 and 1, and a referee who wishes to establish which of the two indeed wins the game. In our initial approach, the referee memorises the entire history of the game. But as we have seen, the occurrence of a cycle in a path-forming game on $\mathcal{G}$ can be detected by storing at most $\text{ent}(\mathcal{G})$ many positions. Hence, if we could provide the referee with the power of sufficiently many detectives, this would reduce the space requirement. The crux of the matter is how to fit such a three-player setting into the two-player model of alternating computation.

Our proposal to overcome this difficulty is to let one of the players act as a referee who challenges the other player in the parity game, but in the same time controls the detectives in an overlying thief and detective game which regards the interactively formed path as if it would be formed by the thief alone.

Formally, this leads to a new game. For a game graph $\mathcal{G} = (V, V_0, E, \Omega)$, a player $i \in \{0,1\}$, and a number k, the *superdetective* game $\mathcal{G}[i,k]$ is played between the Superdetective controlling k detectives and the positions of V_i, and the Challenger in hold of the positions in V_{1-i}. Starting from an initial position position v_0, in any move the Superdetective may place one of the k detectives on the current position v, or leave them in place. If the current position v belongs to V_{1-i}, Challenger has to move to some position $w \in vE$, otherwise the Superdetective moves. (If a player gets stuck, he immediately loses.) The play ends if a position w occupied by a detective is reached and the Superdetective wins if, and only if, the least priority seen since the detective was placed there is even, for $i = 0$ respectively odd, for $i = 1$.

The following lemma states that parity games can be reduced to Superdetective games with an appropriate number of detectives.

Lemma 20. (1) *If Player i has a winning strategy for the parity game $\mathcal{G}$, then the Superdetective wins the superdetective game $\mathcal{G}[i,k]$ with $k = \mathrm{ent}(\mathcal{G})$.*
(2) *If for some $k \in \mathbb{N}$, the Superdetective wins the game $\mathcal{G}[i,k]$, then Player i has a winning strategy for the parity game $\mathcal{G}$.*

Proof. Let σ be a memoryless winning strategy of Player i for the game $\mathcal{G}$ and let $\mathcal{G}_\sigma$ be the subgame of $\mathcal{G}$ induced by this strategy. Then, the least priority seen on any cycle of $\mathcal{G}_\sigma$ is favourable to Player i. This remains true for any cycle formed in $\mathcal{G}[i,k]$ where Player i acting as a Superdetective follows the same strategy σ. On the other hand, obviously $\mathrm{ent}(\mathcal{G}_\sigma) \leq \mathrm{ent}(\mathcal{G}) = k$, which means that the Superdetective also has a strategy to place the k detectives so that every path through $\mathcal{G}_\sigma$ will finally meet a guarded position v and hence form a cycle, witnessing that he wins. This proves (1).

For (2) assume that Player $1-i$ has a memoryless winning strategy τ in the parity game $\mathcal{G}$. But then he could follow this strategy when acting as a Challenger in the $\mathcal{G}[i,k]$, so that the play would actually remain in $\mathcal{G}_\tau$ where no cycle is favourable to Player i. Hence, regardless of the number of detectives, the Superdetective cannot win $\mathcal{G}[i,k]$. □

Note that computing the winner of a superdetective game $\mathcal{G}[i,k]$ requires alternating space $(2k+1)\log|V|$. Indeed, one just plays the game recording the current position of the thief, and the current position of each detective along with the minimal priority that has been seen since he was last posted.

```
procedure Superdetective($\mathcal{G}, v_0, j, k$)
input parity game $\mathcal{G} = (V, V_0, E, \Omega)$, initial position $v_0 \in V$, player $j$, $k$ detectives
// accept iff Superdetective has a winning strategy in $\mathcal{G}[j,k]$ with $k$ detectives
$v := v_0$                                    // current position
$(d_i)_{i \in [k]} := \bot$                   // positions guarded by detectives
$(h_i)_{i \in [k]} := \bot$                   // most significant priorities
```

```
repeat
  if $j = 0$ then
    existentially guess  $i \in [k] \cup \{\text{pass}\}$   // appoint detective $i$ or pass
  else
    universally choose  $i \in [k] \cup \{\text{pass}\}$   // other player's detective
  if $i \neq$ pass then
    $d_i := v$; $h_i := \Omega(v)$                          // guard current node
  $v :=$ Move($\mathcal{G}, v$)                             // simulate a game step
  forall $i \in [k]$ do                                      // update history
    $h_i := \min(h_i, \Omega(v))$
  repeat
until ( $v = d_i$ for some $i$ )                             // cycle detected
if ($j = 0$ and $h_i$ is even) or ($j = 1$ and $h_i$ is odd) then  accept
else  reject
```

We are now ready to prove that parity games of bounded entanglement can be solved in polynomial time. In fact, we establish a more specific result, taking into account the minimal entanglement of subgames induced by a winning strategy.

Theorem 21. *The winner of a parity game $\mathcal{G} = (V, V_0, E, \Omega)$ can be determined in* ASPACE$(\mathcal{O}(k \log |V|))$, *where k is the minimum entanglement of a subgame $\mathcal{G}_\sigma$ induced by a memoryless winning strategy σ in $\mathcal{G}$.* □

Proof. We first describe the procedure informally, by way of a game. Given a parity game $\mathcal{G} = (V, V_0, E, \Omega)$ and an initial position v_0, each player i selects a number k_i and claims that he has a winning strategy from v_0 such that ent$(\mathcal{G}_\sigma) \leq k_i$. The smaller of the two numbers k_0, k_1 is then chosen to verify that Superdetective wins the game $\mathcal{G}[i, k_i]$. If this is the case the procedure accepts the claim of Player i, otherwise Player $(1 - i)$ is declared the winner.

Here is a more formal description of the procedure:

```
procedure SolveParity($\mathcal{G}, v$)
input parity game $\mathcal{G} = (V, V_0, E, \Omega)$, initial position $v \in V$
// accept iff Player 0 wins the game
existentially guess  $k_0 \leq |V|$
universally choose  $k_1 \leq |V|$
if $k_0 \leq k_1$ then
  if Superdetective($\mathcal{G}, v, 0, k_0$) then  accept
  else  reject
else
  if Superdetective($\mathcal{G}, v, 1, k_1$) then  reject
  else  accept
```

We claim that Player 0 has a winning strategy in a parity game $\mathcal{G}, v$ if, and only if, the alternating procedure ParitySolve($\mathcal{G}, v$) accepts.

To see this, assume that Player 0 has a memoryless winning strategy σ from v. Then, the guess $k_0 :=$ ent$(\mathcal{G}_\sigma)$ leads to acceptance. Indeed, for $k_1 \geq k_0$, Player 0 wins the superdetective game $\mathcal{G}[0, k_0]$ by using the strategy σ as a parity player together with the detective strategy for $\mathcal{G}_\sigma$. On the other hand, for $k_1 < k_0$, the procedure accepts as well, since Player 1 cannot win the superdetective game $\mathcal{G}[1, k_1]$ without having a winning strategy for the parity game. The converse follows by symmetric arguments exchanging the roles of the two players. $\square$

Corollary 22. *Parity games of bounded entanglement can be solved in polynomial time.*

Literature

[1] J. L. Balcazar, J. Diaz, and J. Gabarro, *Structural complexity 2*, Springer-Verlag, 1988.

[2] D. Berwanger, *Games and Logical Expressiveness*, Ph. D. Thesis, RWTH Aachen (2005).

[3] D. Berwanger, E. Grädel, and G. Lenzi, *On the variable hierarchy of the modal mu-calculus*, in Computer Science Logic, CSL 2002, J. Bradfield, ed., vol. 2471 of LNCS, Springer-Verlag, 2002, pp. 352–366.

[4] D. Berwanger and G. Lenzi, *The variable hierarchy of the μ-calculus is strict*, in STACS 2005, Proceedings of the 22nd Symposium on Theoretical Aspects of Computer Science, LNCS, Springer-Verlag, 2005.

[5] A. Emerson and C. Jutla, *Tree automata, mu-calculus and determinacy*, in Proc. 32nd IEEE Symp. on Foundations of Computer Science, 1991, pp. 368–377.

[6] T. Johnson, N. Robertson, P. D. Seymour, and R. Thomas, *Directed tree-width*, J. Comb. Theory Ser. B, 82 (2001), pp. 138–154.

[7] M. Jurdziński, *Small progress measures for solving parity games*, in STACS 2000, 17th Annual Symposium on Theoretical Aspects of Computer Science, Proceedings, vol. 1770 of Lecture Notes in Computer Science, Springer, 2000, pp. 290–301.

[8] D. Kozen, *Results on the propositional μ-calculus*, Theoretical Computer Science, 27 (1983), pp. 333–354.

[9] J. Obdrzalek, *Fast mu-calculus model checking when tree-width is bounded*, in CAV'03, vol. 2725 of LNCS, Springer-Verlag, 2003, pp. 80–92.

[10] P. D. Seymour and R. Thomas, *Graph searching and a min-max theorem for tree-width*, J. Comb. Theory Ser. B, 58 (1993), pp. 22–33.

[11] C. Stirling, *Bisimulation, modal logic and model checking games*, Logic Journal of the IGPL, 7 (1999), pp. 103–124.