**Noname manuscript No.**
(will be inserted by the editor)

# Equilibrium Graphs

**Pedro Cabalar · Carlos Pérez ·
Gilberto Pérez**

Received: date / Accepted: date

**Abstract** In this paper we present an extension of Peirce's existential graphs to provide a diagrammatic representation of expressions in Quantified Equilibrium Logic (QEL). Using this formalisation, logical connectives are replaced by encircled regions (circles and squares) and quantified variables are represented as "identity" lines. Although the expressive power is equivalent to that of QEL, the new representation can be useful for illustrative or educational purposes.

## 1 Introduction

Most efforts in Knowledge Representation (KR) have been traditionally focused on symbolic manipulation and, in particular, on logical formulation. The use of a formal representation is surely convenient for automated reasoning, since computer languages provide nowadays excellent tools for symbolic representation and processing. Unfortunately, something that is simpler for computational treatment is not always necessarily better for human understanding. Educational experiences show that learning and understanding logical notation takes some time and effort for new students. Even for an experienced student, reading a formula that nests different quantifiers, connectives or parentheses may become a difficult task and lead to errors in formal specification.

One alternative to formal languages that is probably closer to human's intuition is the use of graphical or diagrammatic representations. In fact, diagrammatic KR has also been explored and used in different fields of Artificial Intelligence – a prominent example is, for instance, Sowa's *conceptual graphs* [14, 15]. But the use of diagrams for logical representation is older than KR and AI, and actually comes from the very origins of modern philosophical logic. As commented by Sowa in [16], the use of diagrams in logic was something common before the introduction of the

This research was partially supported by MINECO projects TIN2013-42149-P and TIN2017-84453-P, Spain.

Department of Computer Science
University of Corunna, Spain
E-mail: {cabalar,c.pramil,gperez}@udc.es

current notation, conceived by Peano[1] in 1889 [9]. In fact, Frege's original formulation of Predicate Calculus already included some diagrammatic component. But it was Charles Sanders Peirce who first introduced[2] a full-blown non-symbolic system for first-order logic: *existential graphs* [12] (EGs). This graphical system allows a complete characterisation of First-Order Logic in diagrammatic terms, without using logical connectives. However, save for few exceptions (like their influence in Sowa's conceptual graphs [16]), the truth is that EGs did not gain the same popularity as the symbolic notation for classical logic, even though they provide an elegant and simple representation that seems very suitable for educational purposes. Perhaps one of the difficulties for their consolidation has to do with their strong dependence on classical logic. Existential graphs take conjunction, negation and existential quantifiers as primitive constructors, building all the rest (disjunction, implication or universal quantification) as derived operations. This approach leaves no room for other non-classical logics such as intuitionistic or intermediate logics, where we may need to keep all these connectives independently.

In this paper we study an extension of existential graphs to be used as an alternative diagrammatic notation for *Answer Set Programming* [7,8,2] (ASP) and, in particular, for its logical formalisation in terms of *Equilibrium Logic* [10]. Proposed by David Pearce, Equilibrium Logic has allowed one to apply the *stable model* semantics [5], originally defined for the syntax of logic programs, to the case of arbitrary propositional formulas. Moreover, the extension to the first-order case, known as *Quantified Equilibrium Logic* [11], provides nowadays a general logical notion of stable models for arbitrary theories expressed in the syntax of First-Order Logic. Equilibrium Logic is defined by imposing a model selection criterion on top of a monotonic intermediate logic known as the logic of *Here-and-There* [6] (HT). In this logic, implication is a primitive operation and, although disjunction can be defined in terms of the former plus conjunction, its representation as a derived operator is rather cumbersome. Something similar happens in *Quantified HT* [11] where, again, the existential quantifier is definable in terms of the universal one, but it is much more convenient to treat both of them as primitive connectives. In the paper, we extend EGs to allow us to deal with all these operators independently by just adding a new graphical primitive (rectangles) to the closed curves and lines already existing in EGs.

The rest of the paper is organised as follows. In the next section, we provide an overview of Existential Graphs, both *alpha* graphs corresponding to propositional logic, and *beta* graphs for first-order logic. In Section 3, we summarise the main definitions of Quantified Equilibrium Logic, assuming a static Herbrand domain, which is the most common case in ASP. The main contributions are presented in Sections 4 and 5 that respectively introduce the extensions of alpha and beta graphs for Equilibrium Logic. Section 6 provides an example from a well-known problem usually treated in ASP. Finally, Section 7 concludes the paper.

[1] Peano's quantifiers $\exists, \forall$ correspond to the inverted letters E and A, whereas $\vee$ comes from Latin *vel* ("or") and conjunction $\wedge$ from its inversion.

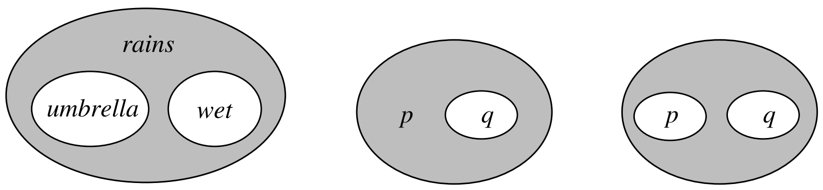
[2] Peirce's first proposals of existential graphs date back to 1882, even earlier than Peano's publication of the modern symbolic notation.

## 2 Existential Graphs

We recall next the essential components of existential graphs. Peirce classified EGs into three types, *alpha*, *beta* and *gamma*, that respectively correspond to Propositional Calculus, First-Order Logic with equality and (a kind of) normal modal logic. We start defining alpha graphs as follows. A *diagram* in alpha graphs is recursively defined as one of the following:

– the main page (when empty, it represents truth)
– atomic propositions
– a region encircled by a closed curve (called *cut*), which denotes the negation of the subdiagram inside the region. An empty cut represents falsity.
– finally, although it is not a drawing in itself, the inclusion of several elements inside the same region or cut (including the full page) is implicitly understood as their conjunction

As an example, Fig. 1(a) explicitly represents the formula $\neg(rains \wedge \neg umbrella \wedge \neg wet)$ which can also be seen as the implications $rains \wedge \neg umbrella \rightarrow wet$ or $rains \wedge \neg wet \rightarrow umbrella$ or the disjunction $\neg rains \vee umbrella \vee wet$, since all these representations are equivalent in classical propositional logic. Using conjunction and negation as primitive operators, we can easily represent an implication $p \rightarrow q$ as $\neg(p \wedge \neg q)$ (Fig. 1(b)) and a disjunction $p \vee q$ as $\neg(\neg p \wedge \neg q)$ (Fig. 1(c)). Another common feature shown in these examples is that areas encircled by an odd number of cuts (negative areas) are sometimes shaded to facilitate the visualisation.



(a) An example (b) $p \rightarrow q$ (c) $p \vee q$

**Fig. 1** Some alpha graphs.

The alpha system was accompanied by a set of inference and equivalence (diagram redrawing) rules that was proved to be sound and complete with respect to Propositional Calculus (note that, at the time, Tarskian model-based semantics had not been developed yet). In this paper, we will focus on the representation itself, leaving graphical inference in the logic of HT or even in (non-monotonic) Equilibrium Logic for a future study.

For representing first-order expressions, Peirce extended alpha graphs to *beta* graphs by the inclusion of a new type of component in the diagram, *lines of identity*. A line of identity is an open line that connects one or more atom names. When it is used to connect more than two atom names, the identity line may bifurcate as many times as needed, getting the shape of a tree or a spider with several ramifications. The reading for an identity line is an existential quantifier: "there exists some individual such that . . . " Figure 2 shows several examples. Fig. 2(a) asserts that there is a red car parked at a street: $\exists x \exists y (car(x) \wedge red(x) \wedge parkedAt(x, y) \wedge street(y))$.

Fig. 2(b) means that there is some person that loves herself, $\exists x(person(x) \wedge loves(x,x))$. Fig. 2(c) says that every man is mortal, $\neg\exists x(man(x) \wedge \neg mortal(x))$ or, if preferred, $\forall x(man(x) \rightarrow mortal(x))$. Finally, Fig. 2(d) specifies that there is a woman adored by every catholic: $\exists x(woman(x) \wedge \forall y(catholic(y) \rightarrow adores(y,x)))$.

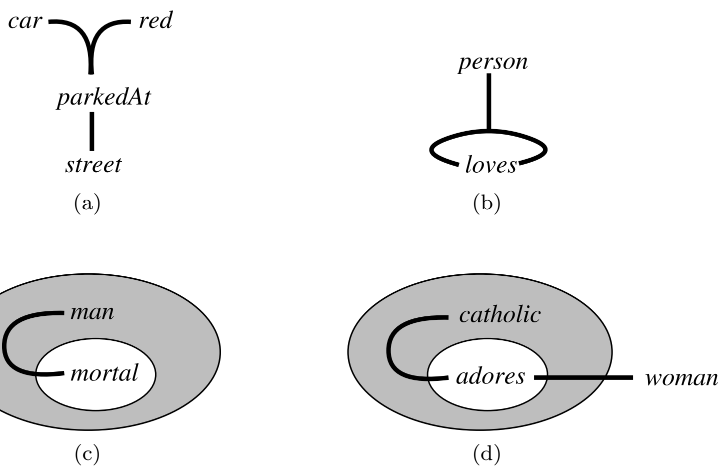


**Fig. 2** Examples of beta graphs.

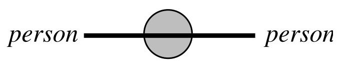
As we can see, identity lines introduce a subtle difference in the role of atom names in beta graphs. Atoms represent now $n$-ary predicates whose arguments correspond to imaginary place holders surrounding the atom name that are used as endpoints of identity lines.[3] In the case of unary predicates, such as *man* or *car*, the position of this place holder is irrelevant. However, when the predicate arity is greater than one, the argument location becomes relevant: for instance, in Fig. 2(d), predicate *adores* has a left argument that corresponds to the adorer and a right argument corresponding to the adored person. In the rest of the paper, we assume that outgoing lines from a predicate name correspond to arguments ordered by the clockwise sequence: *left* < *up* < *right* < *down*.

Another important observation is that beta graphs do not provide a specific method for representing constants. For instance, there is no way for expressing that every catholic adores (Virgin) Mary other than using a unary predicate *Mary* to designate that specific person instead of some abstract *woman*.

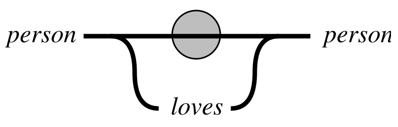
One final remark on identity lines is that they can be actually seen as an implicit equality predicate. Some representations even introduce a label "is" for the identity line to emphasize this feature. Following this interpretation, when an identity runs through an empty cut we get a convenient way to represent an inequality of the form $x \neq y$. Figure 3(a) is a straightforward example of inequality, meaning that there exist (at least) two different persons, while Figure 3(b) is an elaboration that further asserts that one of these two (different) persons love each other. Note

[3] Note that identity lines must be contiguous: the never "follow through" an atom name. For instance, Fig. 2(a) contains *two* identity lines: the bifurcated one on top of *parkedAt* and the single one below.

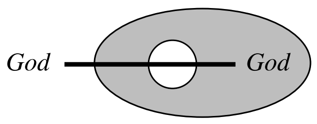
that the same line may be used now to represent two or more different individuals and this feature can be combined in nested cuts. For instance, the diagram in Figure 3(c) represents the sentence "there is a God and only one God."



(a) $\exists x \exists y(person(x) \wedge person(y) \wedge x \neq y)$



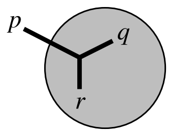
(b) $\exists x \exists y(person(x) \wedge person(y) \wedge loves(x, y) \wedge x \neq y)$



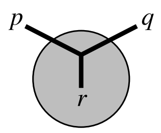
(c) $\exists x(God(x) \wedge \neg \exists y(x \neq y \wedge God(y)))$

**Fig. 3** Three examples involving inequalities.

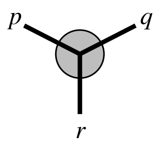
Keeping track of the implicit variables represented by one or more identity lines affected by inequality may become rather cumbersome. In fact, there is no unanimous agreement on this aspect in the literature about EGs, especially when diagrams contain identity lines with ramifications and possibly nested cuts. In our case, we adopt the reading by Shin [13], which we find more suitable for an automated interpretation and a better extension to HT and Equilibrium Logic. Shin's treatment of inequalities can be better illustrated with some examples (we will provide a more formal treatment later on). For instance, Figure 4 shows a ramified identity line connecting three predicates, $p, q$ and $r$, that is combined with a cut in three different ways (note that the cut does not need to be empty to induce an inequality, but we use empty cuts for the sake of clarity). We can see each crossing of the identity line with the ellipse border as a way of "naming" a (different) individual, and so, requiring a new existentially quantified variable. Accordingly, Figures 4(a), (b) and (c) deal with one, two and three variables, respectively, as it can be observed in their corresponding formulas also shown in the figure. Note how the connections inside the cut are translated as multiple equality predicates, such as $x = y \wedge x = z \wedge y = z$ in Figure 4(c) that represents the inner connection among the three crossing points that occurs inside the circle. These equality predicates are then affected by negation, as any other element that we had included in the cut.



(a) $\exists x(p(x) \wedge \neg(q(x) \wedge r(x)))$



(b) $\exists x \exists y(p(x) \wedge q(y) \wedge \neg(r(x) \wedge x = y))$



(c) $\exists x \exists y \exists z(p(x) \wedge q(y) \wedge r(z) \wedge \neg(x = y \wedge x = z \wedge y = z))$

**Fig. 4** Three variants of ramified identity lines and inequality.

## 3 Quantified Equilibrium Logic

For the sake of completeness, we recall in this section the basic definitions of Quantified Equilibrium Logic for function-free theories and Herbrand domains, since this is the most frequent situation in ASP. We consider first-order languages $\mathcal{L} = \langle D, P \rangle$ built over a set of *constant* symbols, $D$ (the Herbrand domain), and a set of *predicate* symbols, $P$. The sets of $\mathcal{L}$-formulas, $\mathcal{L}$-sentences and atomic $\mathcal{L}$-sentences are defined in the usual way. If $D$ is a non-empty set, we denote by $At(D, P)$ the set of ground atomic sentences of the language $\langle D, P \rangle$. By an $\mathcal{L}$-interpretation $I$ over a set $D$ we mean a subset of $At(D, P)$. A *classical* Herbrand $\mathcal{L}$-structure can be regarded as a tuple $\mathcal{M} = \langle D, I \rangle$ where $I$ is an $\mathcal{L}$-interpretation over $D$.

A *here-and-there $\mathcal{L}$-structure* is a tuple $\mathcal{M} = \langle D, I_h, I_t \rangle$ where $\langle D, I_h \rangle$ and $\langle D, I_t \rangle$ are classical Herbrand $\mathcal{L}$-structures such that $I_h \subseteq I_t$. We say that the structure is *total* when $I_h = I_t$. We can think of a here-and-there structure $\mathcal{M}$ as similar to a first-order classical model, but having two parts, or components, $h$ and $t$, that correspond to two different points or "worlds", 'here' and 'there', in the sense of Kripke semantics for intuitionistic logic, where the worlds are ordered by $h \leq t$.

We assume that $\mathcal{L}$ contains the constants $\top$ and $\bot$ and regard $\neg\varphi$ as an abbreviation for $\varphi \rightarrow \bot$. Satisfaction of formulas is defined as follows. Given some world $w \in \{h, t\}$:

- $\mathcal{M}, w \models \top$, $\mathcal{M}, w \not\models \bot$
- $\mathcal{M}, w \models p$ iff $p \in I_w$ for any atom $p \in At(D, P)$
- $\mathcal{M}, w \models c = d$ iff $c$ and $d$ denote the same constant from $D$
- $\mathcal{M}, w \models \varphi \wedge \psi$ iff $\mathcal{M}, w \models \varphi$ and $\mathcal{M}, w \models \psi$.
- $\mathcal{M}, w \models \varphi \vee \psi$ iff $\mathcal{M}, w \models \varphi$ or $\mathcal{M}, w \models \psi$.
- $\mathcal{M}, t \models \varphi \rightarrow \psi$ iff $\mathcal{M}, t \not\models \varphi$ or $\mathcal{M}, t \models \psi$.
- $\mathcal{M}, h \models \varphi \rightarrow \psi$ iff $\mathcal{M}, t \models \varphi \rightarrow \psi$ and $\mathcal{M}, h \not\models \varphi$ or $\mathcal{M}, h \models \psi$.
- $\mathcal{M}, w \models \forall x \varphi(x)$ iff $\mathcal{M}, w \models \varphi(d)$ for all $d \in D$.
- $\mathcal{M}, w \models \exists x \varphi(x)$ iff $\mathcal{M}, w \models \varphi(d)$ for some $d \in D$.

We say that $\mathcal{M}$ is a *model* of a sentence $\varphi$ iff $\mathcal{M}, h \models \varphi$. The resulting logic is called *Quantified Here-and-There Logic with static domains and decidable equality* (QHT, for short).

**Definition 1 (Equilibrium model)** Let $\varphi$ be an $\mathcal{L}$-sentence. An *equilibrium* model of $\varphi$ is a total model $\mathcal{M} = \langle D, I_t, I_t \rangle$ of $\varphi$ such that there is no model of $\varphi$ of the form $\langle D, I_h, I_t \rangle$ where $I_h$ is a proper subset of $I_t$.

When $\langle D, I_t, I_t \rangle$ is an equilibrium model of $\varphi$ we say that the classical (Herbrand) interpretation $\langle D, I_t \rangle$ is a *stable model*[4] of $\varphi$.

## 4 Equilibrium Alpha Graphs

Let us begin considering the use of alpha graphs to represent Equilibrium Logic theories (or ASP logic programs). A first difficulty we face is that implication is a primitive operator in HT, and cannot be represented in terms of conjunction and disjunction (see Theorem 4 in [1]). This generates a conflict with the use of material implication in alpha graphs, defined in terms of negation and conjunction. To overcome this problem, we replace the cut component (negation) by a new diagrammatic construction we will simply call *conditional*. A conditional has the form of a closed curve (or ellipse) and may contain inside a number $n \geq 0$ of rectangles we call *consequents*. Intuitively, when all the elements inside the ellipse (but not in the rectangles) hold then one of the rectangles must hold (that is, we implicitly have a disjunction of consequents). As an example, Fig. 5(a) represents the implication $toss \rightarrow head \vee tails$. The case of 0 rectangles corresponds to an implication with $\bot$ (the empty disjunction) as a consequent. In other words, a conditional without consequents is just read as a negation, as happens in Peirce's alpha diagrams. As an example, Fig. 5(b) represents now the implication $rains \wedge \neg umbrella \rightarrow wet$, that is, $rains \wedge (umbrella \rightarrow \bot) \rightarrow wet$. It is perhaps worth to compare to Fig. 1(a) where, as we commented before, there was no way to differentiate between a negative condition in the antecedent and a positive condition in the consequent (*wet* and *umbrella* played the same role). This reflected the non-directional nature of material implication. Under our new notation, Fig. 5(b) allows now distinguishing the elements in the consequent (*wet* is inside a rectangle) from those in the antecedent, either negated (*umbrella*) or not (*rains*).

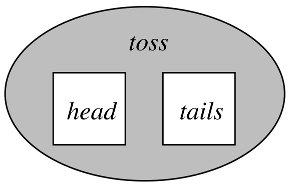
As we have seen, when the conditional has no consequents, it corresponds to a negation. In an analogous way, when the conditional has an an empty antecedent (it only contains rectangles) it obviously represents a disjunction. Fig. 5(c) represents the disjunction $red \vee orange \vee green$ for the possible colors of a traffic light.

A disjunction $p \vee q$ in HT can be defined in terms of conjunction and implication, as it is equivalent to the expression

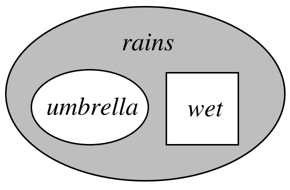
$$((p \rightarrow q) \rightarrow q) \wedge ((q \rightarrow p) \rightarrow p)$$

whose diagrammatic representation is shown in Figure 6. However, the only "advantage" we would gain using this representation (as primitive for disjunction) is that we would not need more than one rectangle in each conditional, while we would clearly lose readability.

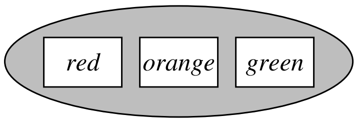
[4] QHT and equilibrium models of first order theories were first defined in [11]. An alternative characterisation of stable models based on second order logic can also be found in [4].



(a) $toss \rightarrow head \vee tails$



(b) $rains \wedge \neg umbrella \rightarrow wet$



(c) $red \vee orange \vee green$

**Fig. 5** Examples of conditionals.

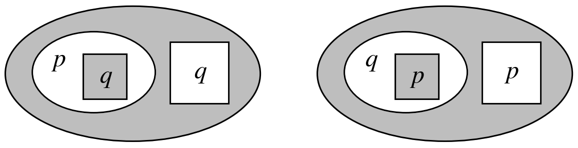
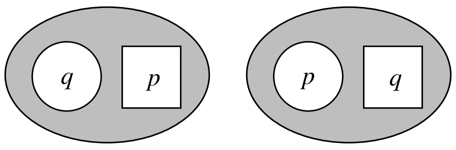
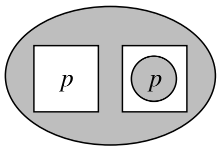


**Fig. 6** $((p \rightarrow q) \rightarrow q) \wedge ((q \rightarrow p) \rightarrow p)$

An interesting construction in ASP is the use of choice rules. The original way to build a choice that causes the non-deterministic addition of an atom $p$ in ASP was using some auxiliary predicate $q$ and building an even negative cycle as the one shown in Figure 7(a). A second possibility that does not require an auxiliary predicate is using the formula $p \vee \neg p$ (which is not a tautology in HT) represented in Figure 7(b).



(a) $(\neg q \rightarrow p) \wedge (\neg p \rightarrow q)$



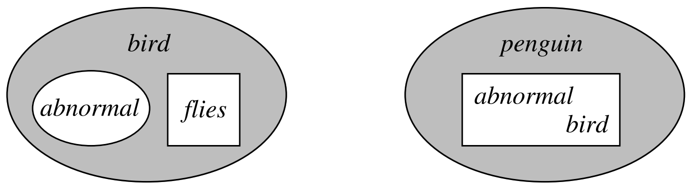
(b) $p \vee \neg p$

**Fig. 7** Choice rules.

To conclude this section, we illustrate a typical example from Non-Monotonic Reasoning. Fig. 8 encodes a propositional program with two rules respectively asserting that a bird normally flies and that a penguin is an abnormal bird.

## 5 Equilibrium Beta Graphs

As happened with implication in the propositional case, the universal quantifier is a primitive operator in QEL and cannot be represented in terms of existential



(a) $bird \wedge \neg abnormal \rightarrow flies$ (b) $penguin \rightarrow abnormal \wedge bird$

**Fig. 8** Birds and penguins.

quantifiers and the other connectives[5]. Therefore, introducing identity lines would not suffice to cover the expressive power of QEL if they were always read as existential quantifiers. Fortunately, since we count with a new conditional connective, whose expressiveness is richer than the simple cut, we can use it to cover both existential and universal quantifiers as follows.

**Definition 2 (Universally Quantified Variable)** A *universally quantified variable* corresponds to any identity line satisfying the next three conditions:

1. It is completely encircled by a conditional (an ellipse) and
2. it has some portion inside the conditional consequent (inner rectangle) and
3. it also has some portion outside the conditional consequent. □

Figure 9 shows some examples combining conditionals and identity lines. Fig. 9(a) corresponds to a universal quantifier, saying that all men are mortal:

$$\forall x(man(x) \rightarrow mortal(x)) \tag{1}$$

Note the difference with respect to the version in Fig. 2(c), where we had a cut (negation) instead of the rectangle. This version is still a correct equilibrium beta graph, but its reading corresponds to:

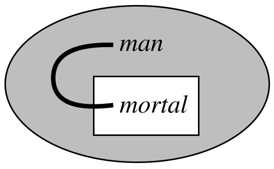
$$\forall x(man(x) \wedge \neg mortal(x) \rightarrow \bot) \tag{2}$$

which has a quite different meaning from (1): the latter is a rule that allows deriving $mortal(x)$ from any $man(x)$, whereas (2) acts as a constraint, forbidding stable models where some man is not known to be mortal. Another equivalent reading of a constraint like Fig. 2(c) (i.e. a conditional with existential lines but no consequents) is just as a negation of an existential quantifier:

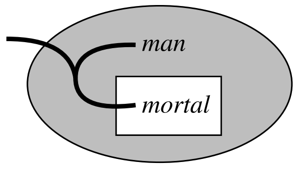
$$\neg \exists x(man(x) \wedge \neg mortal(x))$$

Fig. 9(b) corresponds to an existential quantifier: it contains an identity line which is not encircled by the conditional (it "comes from outside"). As for Fig. 9(c), it represents an existential quantifier: it is encircled by the ellipse, has some portion outside the consequent but it has *no portion* inside the consequent. However, the corresponding existentially quantified formula $\exists x\ man(x)$ is completely inside the antecedent of $(\exists x\ man(x)) \rightarrow mortal$ and this is QHT-equivalent to the universally quantified implication $\forall x(man(x) \rightarrow mortal)$.

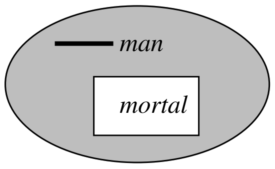
[5] It is actually the other way around. Any existentially quantified formula $\exists x P(x)$ is QHT equivalent to $\forall x \forall y((P(x) \rightarrow P(y)) \rightarrow P(y))$.



(a) $\forall x(man(x) \rightarrow mortal(x))$



(b) $\exists x(man(x) \rightarrow mortal(x))$



(c) $\forall x(man(x) \rightarrow mortal)$

**Fig. 9** Some conditionals with identity lines.

Identity lines in Equilibrium Graphs may also induce an implicit equality predicate, as happened in Peirce's EGs. For instance, Figure 3(a) is also a valid Equilibrium Graph whose symbolic representation would strictly correspond to $\exists x \exists y \ (person(x) \ \wedge \ person(y) \ \wedge \ (x = y \rightarrow \bot))$. The case where an identity line goes through a rectangle crossing at two points $x$ and $y$ should be understood as an equality $x = y$ in the consequent. As an example of this, Figure 10 means that a person's name is unique:

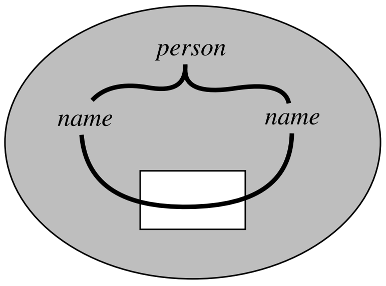


**Fig. 10** $\forall x \forall y(\exists z(person(z) \wedge name(z, x) \wedge name(z, y)) \rightarrow (x = y))$

The general translation of an arbitrary diagram into a first-order formula can be done by a recursive analysis using the concept of *subgraph*.

**Definition 3 (Subgraph)** A *subgraph* $G_i$ of an Equilibrium Beta Graph $G$ is a portion fully enclosed by a closed *boundary*, that is, a cut or a rectangle in $G$. Moreover, a subgraph $G_i$ is said to be *maximal* in $G$ if it is not included inside another subgraph of $G$. □

Figure 11 shows a decomposition of Fig. 9(b) into subgraphs. The initial graph, shown in the upper left corner, can also be seen as the graph to its right, where the maximal subgraph encircled by the cut has been named as $G1$. The content of $G1$ is shown below, in the next line. Then, following again to the right, we identify the maximal subgraph of $G1$ (the rectangle) naming it as $G1.1$ and showing its content below. It is important to note that a subgraph alone is not an Equilibrium Beta graph in a strict sense, since some of its identity lines may be crossing from outside the boundary, so only a segment of the line lies inside. When this happens, we depict the identity line with a dashed ending to emphasize that it comes from outside and call this an *outer ending*.

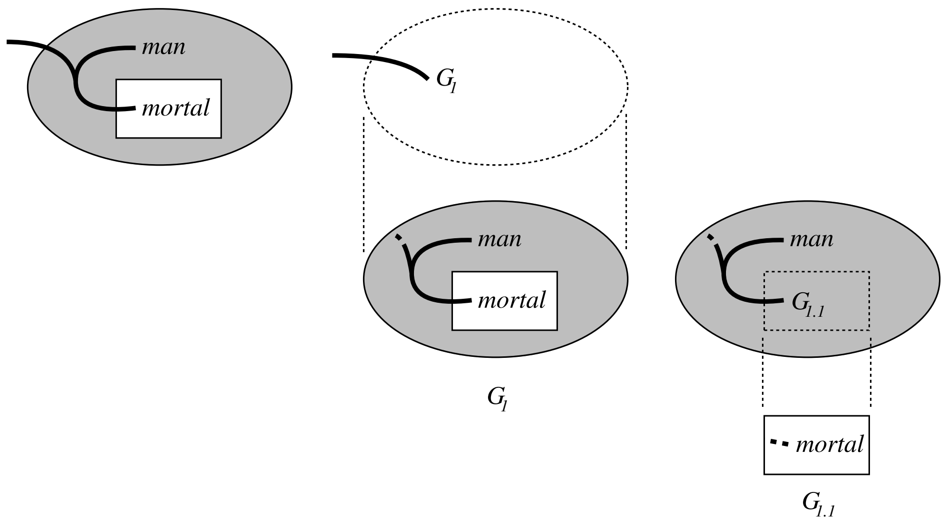


**Fig. 11** An example of decomposition into subgraphs.

The translation of a graph $G$ into a formula $\Phi(G)$ is described by Algorithm 1. Using this algorithm, the formulas we obtain from Figure 11 correspond to the sequence:

$$G \equiv \exists x G_1(x)$$

$$G_1(x) \equiv man(x) \rightarrow G_{1.1}(x)$$

$$G_{1.1}(x) \equiv mortal(x)$$

that, after replacing each predicate $G_i$ by its formula, leads to $\exists x(man(x) \rightarrow mortal(x))$. For a more elaborated example, take the graph $G$ in Figure 12. We begin at the outermost level (the main page) identifying the maximal subgraphs as predicates $G_1$, $G_2$ and $G_3$ shown in Figure 13. This new diagram has a straightforward translation as the formula (3).

$$\begin{aligned} G \equiv\ & \exists x_1 \exists x_2 \exists x_3 \exists x_4 \\ & (p(x_1) \wedge G_1(x_1,x_2) \wedge G_2(x_2) \wedge q(x_3) \wedge r(x_4) \wedge G_3(x_2,x_4,x_3)) \end{aligned} \tag{3}$$

We proceed then into each subgraph, repeating the process, as shown in Figure 14. The formulas we obtain in this step correspond to:

**Algorithm 1** Formulate($G$)

```
Require: An Equilibrium beta graph or subgraph $G$.
Ensure: It returns the first order formula $\Phi(G)$ corresponding to $G$

  $G'$ := result of replacing each maximal subgraph in $G$ by a new predicate name $G_i$
  for all identity lines $L$ occurring in $G'$ do
    if $L$ is completely inside $G'$ (it has no outer endings) then
      Label $L$ with a new fresh variable $x$
      if $G'$ is an ellipse, so $\Phi(G)$ has the form $\alpha \rightarrow \beta$ then
        if $L$ is connected to a subgraph $G_i$ with a rectangular boundary then
          Add prefix $\forall x$ to $\Phi(G)$
        else
          Add prefix $\exists x$ to $\alpha$
        end if
      else
        Add prefix $\exists x$ to $\Phi(G)$
      end if
    else
      {$L$ has outer endings and their variable names were defined at previous steps}
      for all pairs of outer endings of $L$ labelled with names $x$ and $y$ do
        Add the equality $x = y$ as a conjunct in $\Phi(G)$
      end for
    end if
  end for
  Build the rest of $\Phi(G)$ as in Equilibrium alpha graphs, associating the predicates with their
  corresponding variables
  for all maximal subgraphs $G_i$ do
    $\Phi_i$:=Formulate($G_i$)
    $\Phi(G)$:= result of replacing predicate $G_i$ by $\Phi_i$ in $\Phi(G)$
  end for
  return $\Phi(G)$
```

$$G_1(x_1, x_2) \equiv s(x_1) \rightarrow G_{1,1}(x_1, x_2) \tag{4}$$

$$G_2(x_2) \equiv \exists x_5\ (m(x_5) \wedge G_{2,1}(x_5, x_2)) \rightarrow \bot \tag{5}$$

$$G_3(x_2, x_4, x_3) \equiv (x_2 = x_3) \wedge (x_2 = x_4) \wedge (x_3 = x_4) \rightarrow G_{3,1}(x_2) \tag{6}$$

Note that we do not introduce new quantifiers for identity lines that have a dashed (outer) ending, since their quantification is already defined in an upper level of the formula and their names are also "inherited" from that upper level. Now, let us focus on Figure 14(c): its identity line has more than one outer ending. This must be interpreted as an implicit equality predicate meaning that all the inherited variables are in fact the same logical object inside this subgraph. This explains the three equalities in (6) (actually, one of them is redundant) for $x_2, x_3$ and $x_4$, so that for the free variable of $G_{3,1}$ we could actually choose any of them (in this case, $x_2$). Finally, if we recursively repeat the process for each new subgraph we obtain Figure 15 and the corresponding formulas:

$$\begin{aligned} G_{1,1}(x_1, x_2) &\equiv \exists x_6\ t(x_6) \wedge (x_1 = x_2) \\ G_{2,1}(x_5, x_2) &\equiv (x_5 = x_2) \rightarrow \bot \\ G_{3,1}(x_2) &\equiv n(x_2) \end{aligned}$$

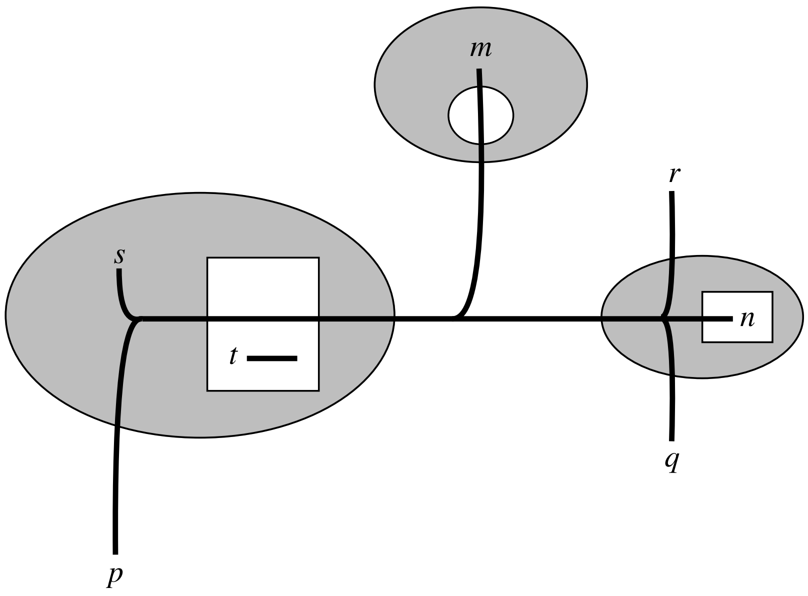


**Fig. 12** A more complex example.

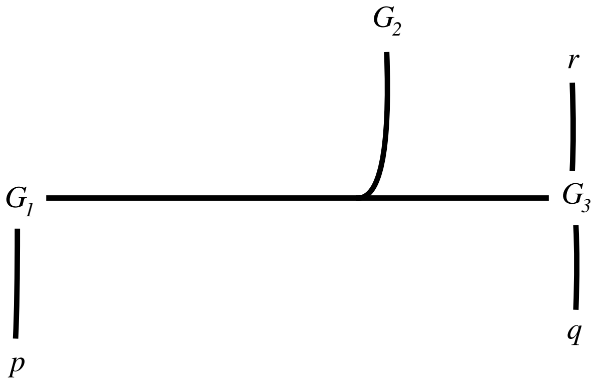


**Fig. 13** Step 1. Subgraphs at first level are replaced by predicates.

After replacing each subgraph by its formula we obtain the final result:

$$\begin{aligned} G \equiv\ & \exists x_1 \exists x_2 \exists x_3 \exists x_4 \\ & \big(\ p(x_1) \wedge q(x_3) \wedge r(x_4) \\ & \wedge (s(x_1) \rightarrow (\exists x_6\ t(x_6) \wedge (x_1 = x_2))) \\ & \wedge (\exists x_5\ (m(x_5)\ \wedge\ ((x_5 = x_2) \rightarrow \bot)) \rightarrow \bot) \\ & \wedge ((x_2 = x_3) \wedge (x_2 = x_4) \wedge (x_3 = x_4) \rightarrow n(x_2))\ \ \big) \end{aligned}$$

The translation we have defined is coherent with Peirce's EGs in the following way. Given an Equilibrium Beta Graph $G$ we may obtain its corresponding

Peirce's Beta Graph $G^*$ by replacing all rectangles in $G$ by ellipses[6]. Then, the symbolic representations of $G$ and $G^*$ are equivalent in classical First-Order Logic, as formally stated below.

**Proposition 1** *Let $\Phi(G)$ denote the first-order formula associated to $G$ under the interpretation in the current paper and $\Phi(G^*)$ the formula associated to $G^*$ under Peirce's beta graphs interpretation. Then $\Phi(G)$ and $\Phi(G^*)$ are equivalent in classical First-Order Logic.* □

For instance, it is easy to see that the formula $\Phi(G)$ in the caption of Figure 10 and the corresponding Peirce's reading $\Phi(G^*)$ of the same diagram:

$$\neg\exists x \exists y \exists z \; (person(z) \wedge name(z,x) \wedge name(z,y) \wedge \neg(x=y))$$

are classically equivalent.

## 6 An example from ASP

In this section we provide an example encoding the well-known Hamiltonian cycle problem: given a graph $G$, find a cyclic path that visits each node in $G$ exactly once. We assume that the graph $G$ is provided in terms of facts for the binary predicate *edge*, related to node names. The Hamiltonian path is encoded using a binary predicate *in*, meaning that the corresponding edge is included in the path, for a given stable model.

Figure 16 shows a possible diagrammatic representation of this problem. The corresponding formulas, reading all the conditionals from left to right and from up to down would respectively be:

$$\forall x \forall y \;\; \big( \; edge(x,y) \rightarrow node(x) \wedge node(y) \; \big) \tag{7}$$
$$\forall x \forall y \;\; \big( \; edge(x,y) \rightarrow in(x,y) \vee \neg in(x,y) \; \big) \tag{8}$$
$$\neg\exists x \exists y \exists z \;\; \big(in(x,y) \wedge in(x,z) \wedge y \neq z \; \big) \tag{9}$$
$$\neg\exists x \exists y \exists z \;\; \big(x \neq y \wedge in(x,z) \wedge in(y,z) \; \big) \tag{10}$$
$$\forall x \forall y \;\; \big( \; in(x,y) \rightarrow reach(x,y) \; \big) \tag{11}$$
$$\forall x \forall y \forall z \;\; \big( \; reach(x,y) \wedge in(y,z) \rightarrow reach(x,z) \; \big) \tag{12}$$
$$\neg\exists x \exists y \;\; \big(node(x) \wedge node(y) \wedge \neg reach(x,y) \; \big) \tag{13}$$

Formula (7) asserts that the two arguments of predicate *edge* are nodes. (8) is a non-deterministic choice to include any edge in the stable model or not. (9) and (10) are constraints respectively forbidding that two edges in the path with the same origin go to two different targets, and vice versa, that two different origin nodes go to a common target. Formulas (11),(12) define the transitive closure *reach* of predicate *in*. Finally, (12) is a constraint forbidding that a node $y$ cannot be reached from another node $x$.

This type of formulas has a quite immediate translation to standard ASP syntax. The theory (7)-(13) can be directly written as the ASP program:

[6] We could alternatively say that $G^*$ is just "Peirce's reading" of $G$ without need of any transformation, since a rectangle is also a case of closed curve, and Peirce's original approach would make no real distinction between ellipses and rectangles.

```
1 node(X) :- edge(X,Y).
2 node(Y) :- edge(X,Y).
3 {in(X,Y)} :- edge(X,Y).
4 :- in(X,Y), in(X,Z), Y!=Z.
5 :- X!=Y, in(X,Z), in(Y,Z).
6 reach(X,Y) :- in(X,Y).
7 reach(X,Z) :- reach(X,Y), in(Y,Z).
8 :- node(X), node(Y), not reach(X,Y).
```

where lines 1,2 correspond to (7) and lines 3-7 respectively correspond to (8)-(13). This type of implications $\alpha \rightarrow \beta$ are called rules and represented from left to right $\beta$ `:-` $\alpha$. Conjunction is replaced by comma and $\neg$ by `not`. As customary in logic programming, variables begin with a capital letter. Formulas with a conjunction in the consequent, such as (7), are separated into different rules (in the example, lines 1 and 2). Finally, the construction $\alpha \vee \neg\alpha$ in a consequent, like $in(x,y) \vee \neg in(x,y)$ in (8), is usually represented as a so-called *choice expression*, $\{\alpha\}$, meaning that $\alpha$ can be included in the answer set or not, in a non-deterministic way. For the sake of simplicity, we do not provide a definition of ASP semantics in this paper, since equilibrium models coincide with answer sets, for the syntactic fragment currently accepted by ASP solvers.

Figure 17 shows three diagrams respectively depicting a possible example of input graph (facts for predicate *edge*) plus the two corresponding stable models that represent the Hamiltonian paths of the input graph.

## 7 Conclusions

By introducing a minimal variation on Peirce's existential graphs (the introduction of rectangles), we have presented a diagrammatic representation of Quantified Equilibrium Logic and ASP programs. In fact, the current formulation allows one to represent any intermediate logic, since it has just allowed defining implication, disjunction and universal quantification as primitive constructions, rather than derived operators in terms of conjunction, negation and existential quantifiers.

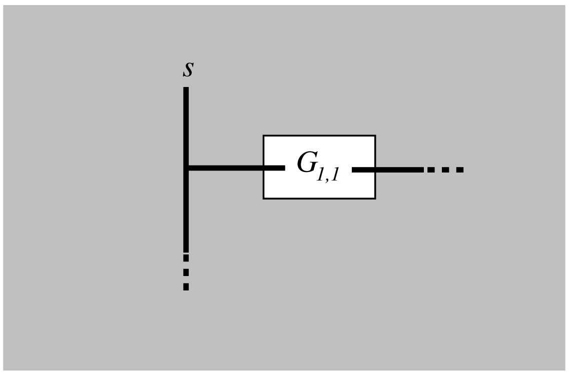
This paper constitutes a first proposal but, obviously, much work is left to do yet. First, it is unclear how to provide a fully visual semantic characterisation, especially for beta diagrams. Another desirable feature would be a set of inference and equivalence diagram-rewriting rules that covered QHT in a sound and complete way. Under the ASP perspective, it is interesting to note that not every QEL formula corresponds to an ASP program: for instance, formulas beginning by existential quantifiers are not ASP representable. It would also be interesting to identify graphical features of the kind of diagrams that have a direct translation into ASP. This also includes the graphical characterisation of *safety* conditions, required for a suitable grounding. The introduction of complex ASP constructs such as aggregates or preferences, or even the diagrammatic representation for arithmetic expressions constitute an important difficulty still to be solved . Finally, regarding implementation, we have developed a prototype called `Grasp`[7] allowing graphical manipulation of Equilibrium Graphs and their interpretation using an

[7] `https://github.com/Minimuino/ASP-Graph`

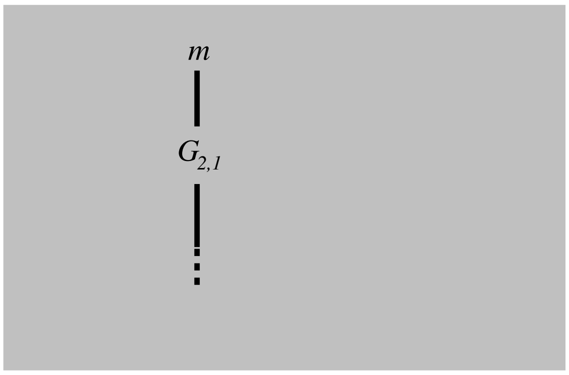
ASP solver as a backend. A promising line to explore would be the integration into a full visual tool for ASP like the one described in [3].

## References

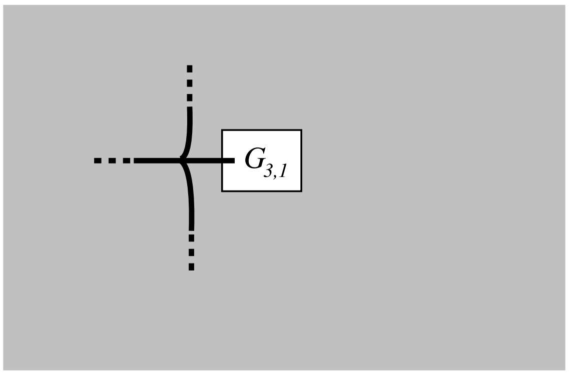
1. F. Aguado, P. Cabalar, D. Pearce, G. Pérez, and C. Vidal. A denotational semantics for equilibrium logic. *Theory and Practice of Logic Programming*, 15(4-5):620–634, 2015.
2. G. Brewka, T. Eiter, and M. Truszczyński. Answer set programming at a glance. *Communications of the ACM*, 54(12):92–103, 2011.
3. Onofrio Febbraro, Kristian Reale, and Francesco Ricca. A visual interface for drawing ASP programs. In Wolfgang Faber and Nicola Leone, editors, *Proceedings of the 25th Italian Conference on Computational Logic, Rende, Italy, July 7-9, 2010*, CEUR Workshop Proceedings. 2010.
4. Paolo Ferraris, Joohyung Lee, and Vladimir Lifschitz. A new perspective on stable models. In Manuela M. Veloso, editor, *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12*, pages 372–379, 2007.
5. M. Gelfond and V. Lifschitz. The Stable Model Semantics For Logic Programming. In *Proc. of the 5th International Conference on Logic Programming (ICLP'88)*, pages 1070–1080, Seattle, Washington, 1988.
6. A. Heyting. Die formalen Regeln der intuitionistischen Logik. *Sitzungsberichte der Preussischen Akademie der Wissenschaften. Physikalisch-mathematische Klasse*, 1930.
7. V. Marek and M. Truszczyński. *Stable models and an alternative logic programming paradigm*, pages 169–181. Springer-Verlag, 1999.
8. I. Niemelä. Logic Programs with Stable Model Semantics as a Constraint Programming Paradigm. *Annals of Mathematics and Artificial Intelligence*, 25(3-4):241–273, 1999.
9. Giuseppe Peano. Aritmetices principia nova methoda exposita, 1889. Torino: Bocca.
10. D. Pearce. A New Logical Characterisation of Stable Models and Answer Sets. In *Proc. of Non-Monotonic Extensions of Logic Programming (NMELP'96)*, pages 57–70, Bad Honnef, Germany, 1996.
11. David Pearce and Agustín Valverde. Quantified equilibrium logic and foundations for answer set programs. In *Proc. of the 24th Intl. Conf. on Logic Programming, ICLP 2008, (Udine, Italy, December 9-13)*, volume 5366 of *Lecture Notes in Computer Science*, pages 546–560. Springer, 2008.
12. Charles Sanders Peirce. Manuscripts on existential graphs. In *Collected Papers of Charles Sanders Peirce*, volume 4, pages 320–410. Harvard University Press, Cambridge, MA, 1906.
13. Sun-Joo Shin. *The Iconic Logic of Peirce's Graphs.* Bradford Book, 2002.
14. John F. Sowa. Conceptual graphs for a data base interface. *IBM Journal of Research and Development*, 20(4):336–357, 1976.
15. John F. Sowa. Conceptual graphs. In Frank van Harmelen, Vladimir Lifschitz, and Bruce W. Porter, editors, *Handbook of Knowledge Representation*, volume 3 of *Foundations of Artificial Intelligence*, pages 213–237. Elsevier, 2008.
16. John F. Sowa. From existential graphs to conceptual graphs. *IJCSSA*, 1(1):39–72, 2013.



(a) $G_1$



(b) $G_2$



(c) $G_3$

**Fig. 14** Step 2.

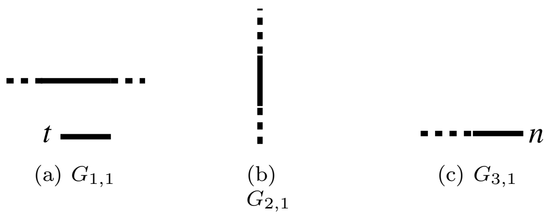


**Fig. 15** Step 3.

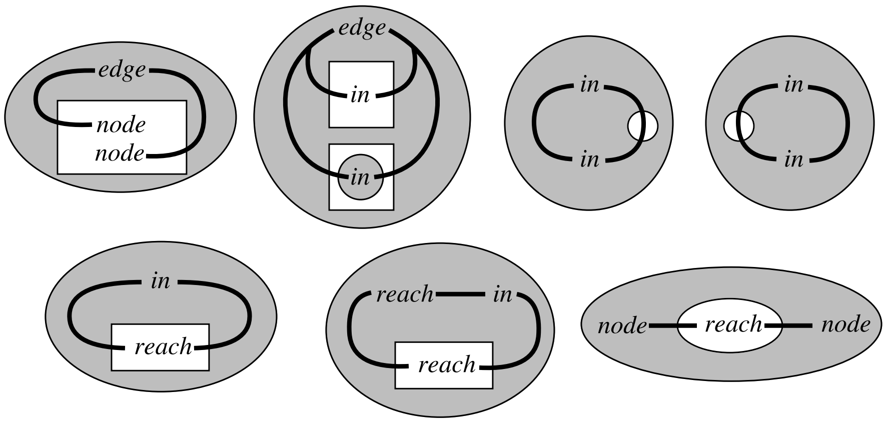


**Fig. 16** Graphical encoding of the Hamiltonian cycles problem.

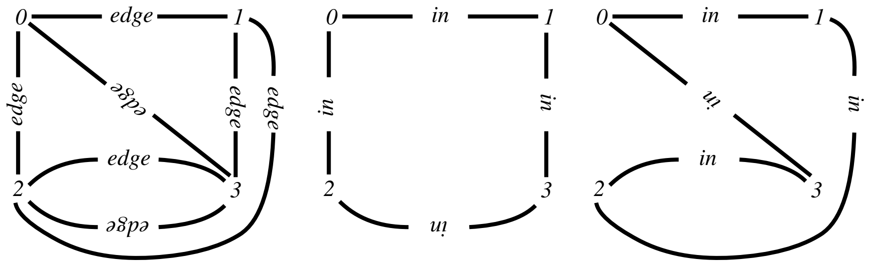


**Fig. 17** A graph and its two stable models corresponding to the Hamiltonian paths.