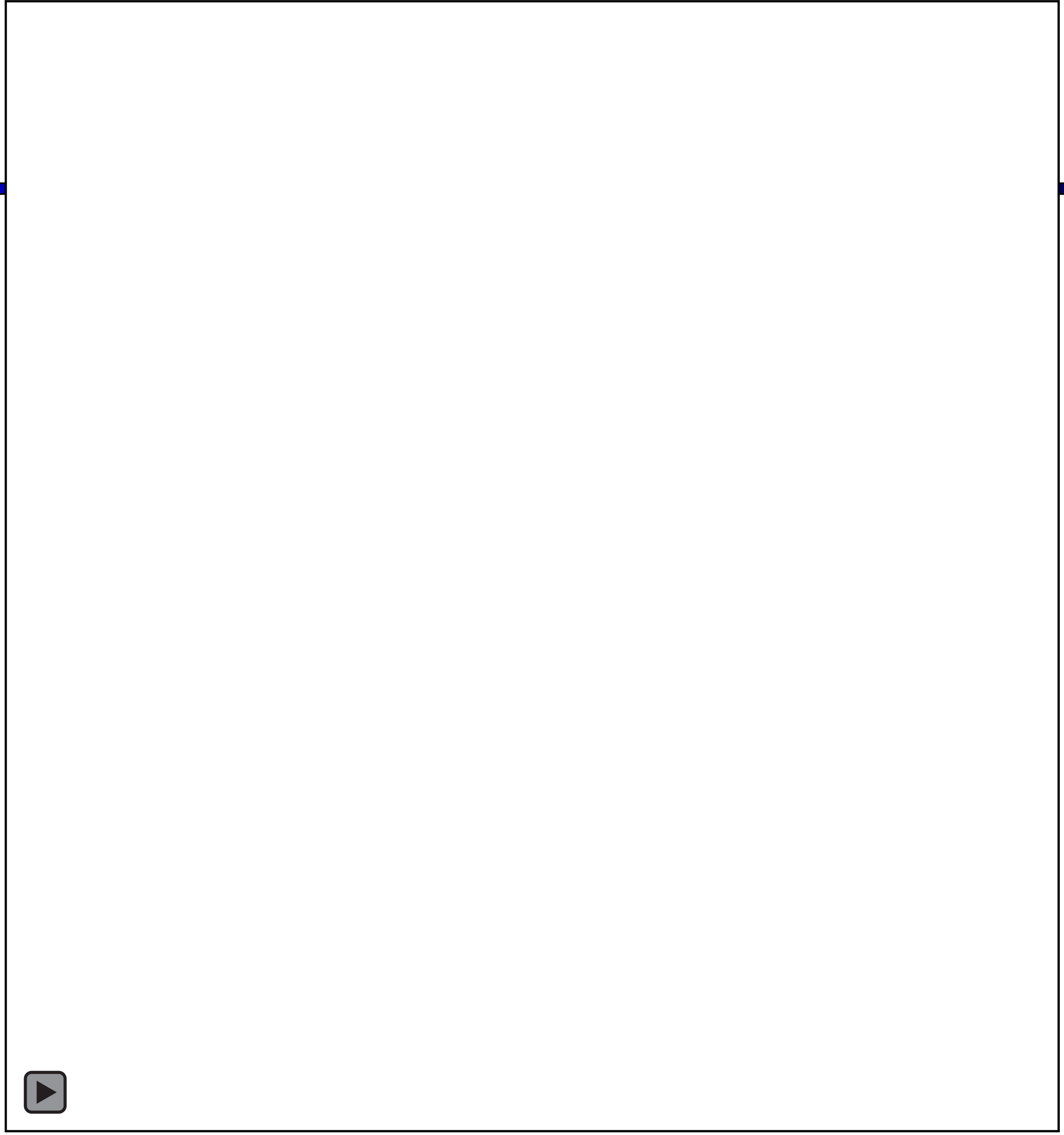
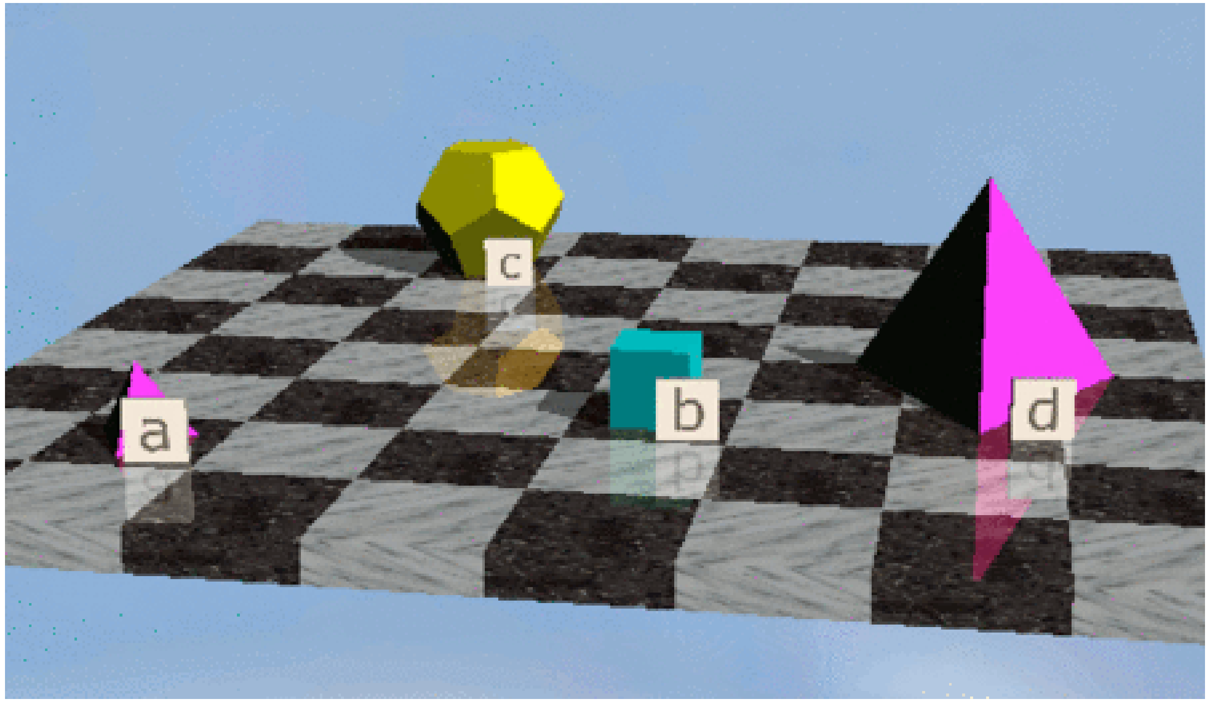


CS 188: Artificial Intelligence

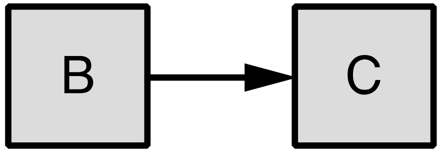
First-Order Logic



Instructor: Stuart Russell

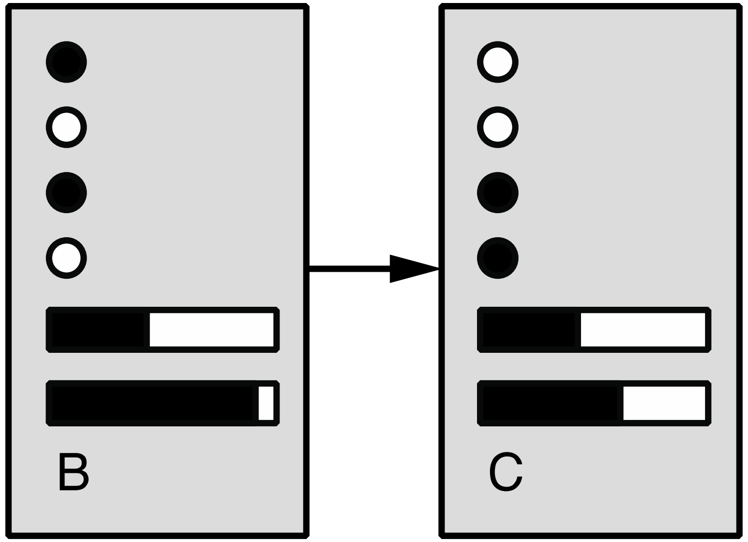
University of California, Berkeley

Spectrum of representations



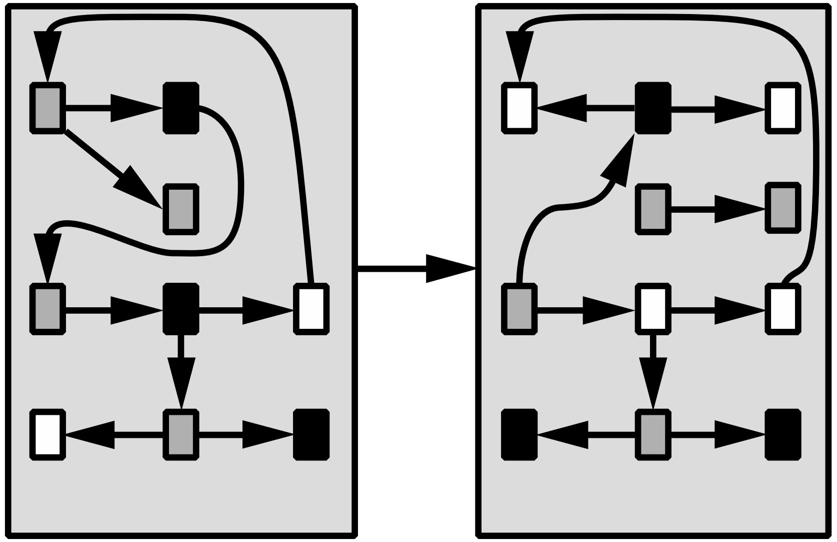
(a) Atomic

Search, game-playing



(b) Factored

CSPs, planning, propositional logic, Bayes nets, neural nets



(b) Structured

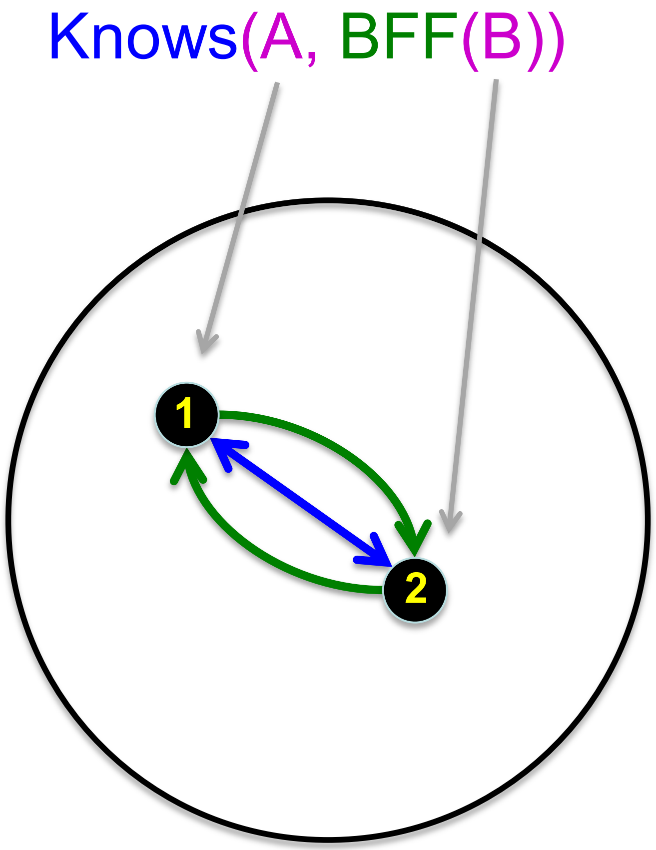
First-order logic, databases, probabilistic programs

Expressive power

- Rules of chess:
  - 100,000 pages in propositional logic
  - 1 page in first-order logic
- Rules of pacman:
  - $\forall x,y,t\ At(x,y,t) \Leftrightarrow [At(x,y,t-1) \wedge \neg\exists\ u,v\ Reachable(x,y,u,v,Action(t-1))]\ \vee$
    $[\exists\ u,v\ At(u,v,t-1) \wedge Reachable(x,y,u,v,Action(t-1))]$

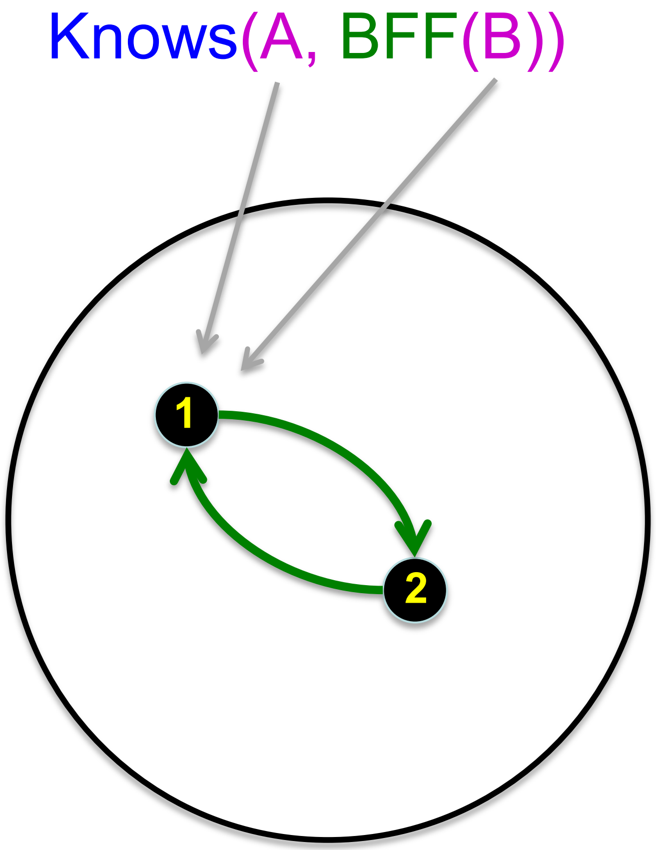
Possible worlds

- A possible world for FOL consists of:
  - A non-empty set of objects
  - For each k-ary predicate in the language, a set of k-tuples of objects (i.e., the set of tuples of objects that satisfy the predicate in this world)
  - For each k-ary function in the language, a mapping from k-tuples of objects to objects
  - For each constant symbol, a particular object (can think of constants as 0-ary functions)



Possible worlds

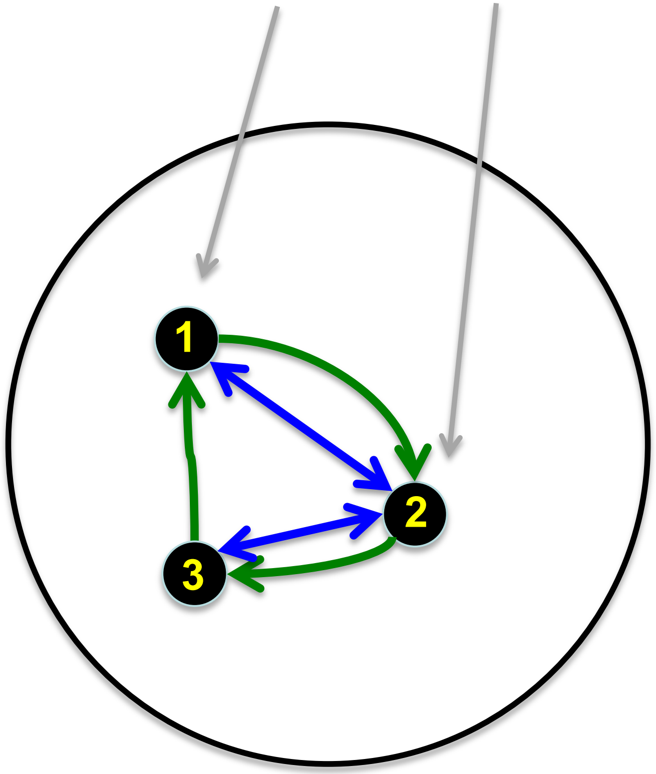
- A possible world for FOL consists of:
  - A non-empty set of objects
  - For each k-ary predicate in the language, a set of k-tuples of objects (i.e., the set of tuples of objects that satisfy the predicate in this world)
  - For each k-ary function in the language, a mapping from k-tuples of objects to objects
  - For each constant symbol, a particular object (can think of constants as 0-ary functions)



Possible worlds

- A possible world for FOL consists of:
  - A non-empty set of objects
  - For each k-ary predicate in the language, a set of k-tuples of objects (i.e., the set of tuples of objects that satisfy the predicate in this world)
  - For each k-ary function in the language, a mapping from k-tuples of objects to objects
  - For each constant symbol, a particular object (can think of constants as 0-ary functions)

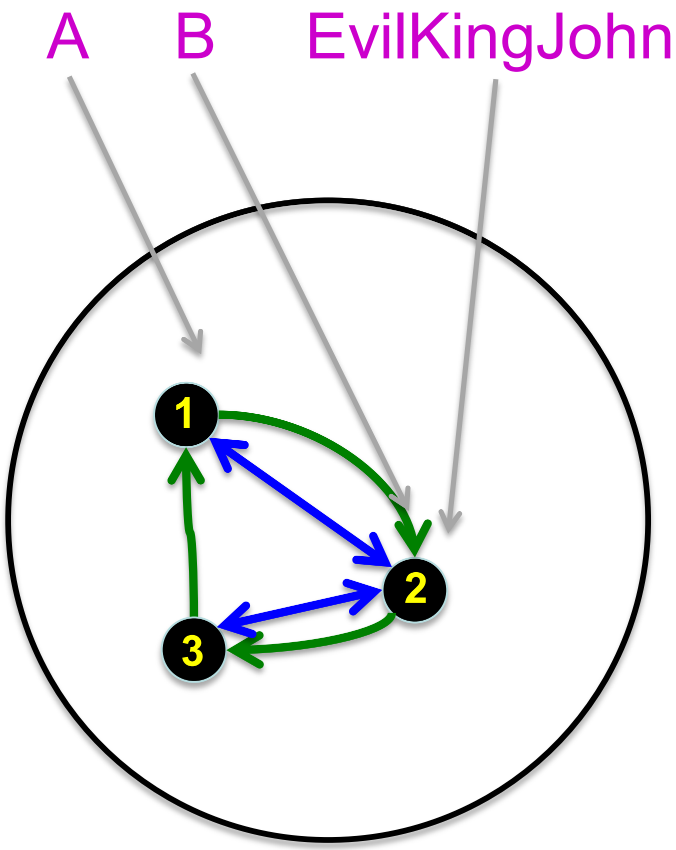
Knows(A, BFF(B))



How many possible worlds?

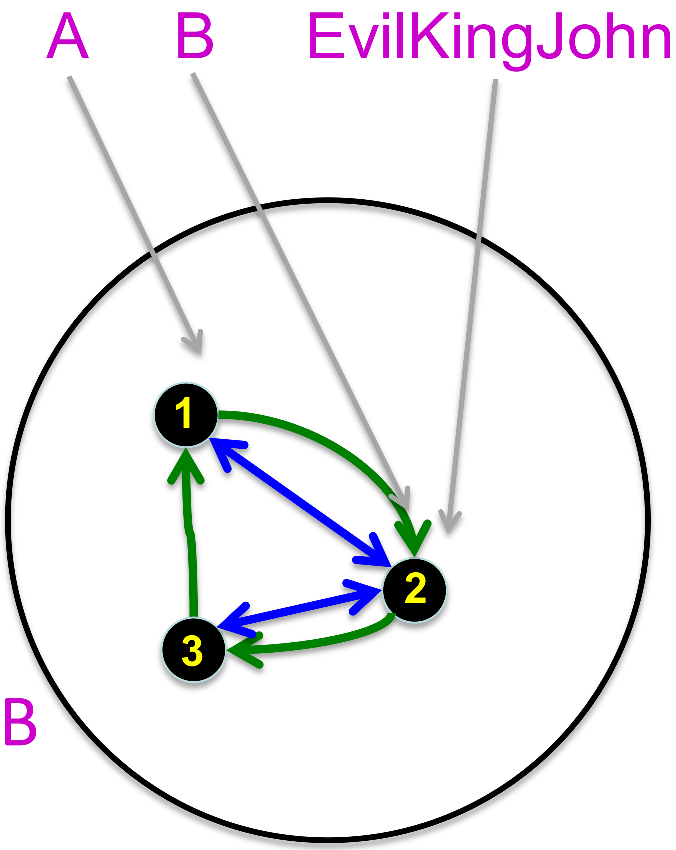
Syntax and semantics: Terms

- A term refers to an object; it can be
  - A constant symbol, e.g., A , B, EvilKingJohn
    - The possible world fixes these referents
  - A function symbol with terms as arguments, e.g., BFF(EvilKingJohn)
    - The possible world specifies the value of the function, given the referents of the terms
      - BFF(EvilKingJohn) -> BFF(**2**) -> **3**
  - A logical variable, e.g., x
    - (more later)



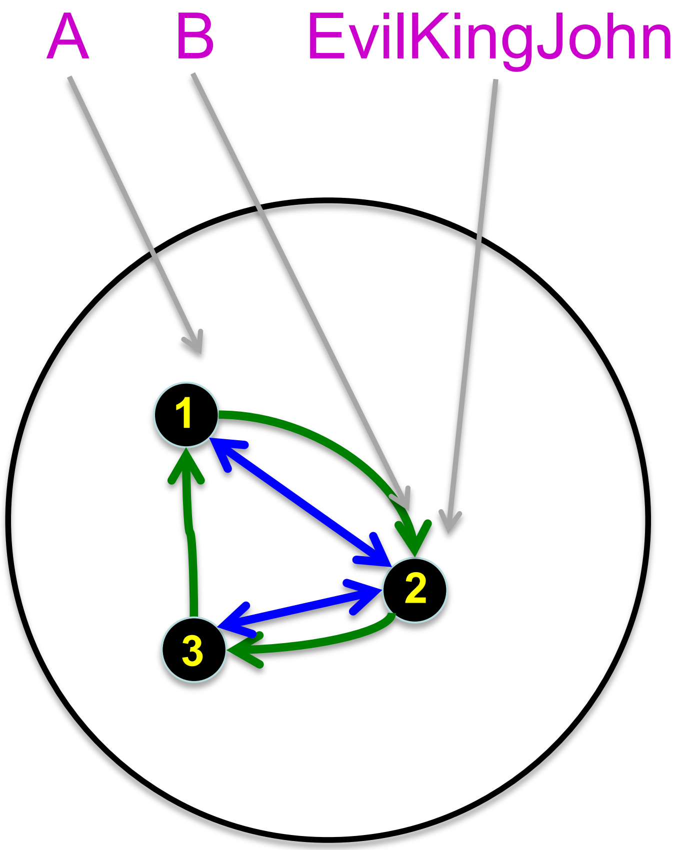
Syntax and semantics: Atomic sentences

- An atomic sentence is an elementary proposition (cf symbols in PL)
  - A predicate symbol with terms as arguments, e.g., Knows(A,BFF(B))
    - True iff the objects referred to by the terms are in the relation referred to by the predicate
    - Knows(A,BFF(B)) -> Knows(**1**,BFF(**2**)) -> Knows(**1**,**3**) -> **F**
  - An equality between terms, e.g., BFF(BFF(BFF(B)))=B
    - True iff the terms refer to the same objects
    - BFF(BFF(BFF(B)))=B -> BFF(BFF(BFF(**2**)))=**2** -> BFF(BFF(**3**))=**2** -> BFF(**1**)=**2** -> **2**=**2** -> **T**



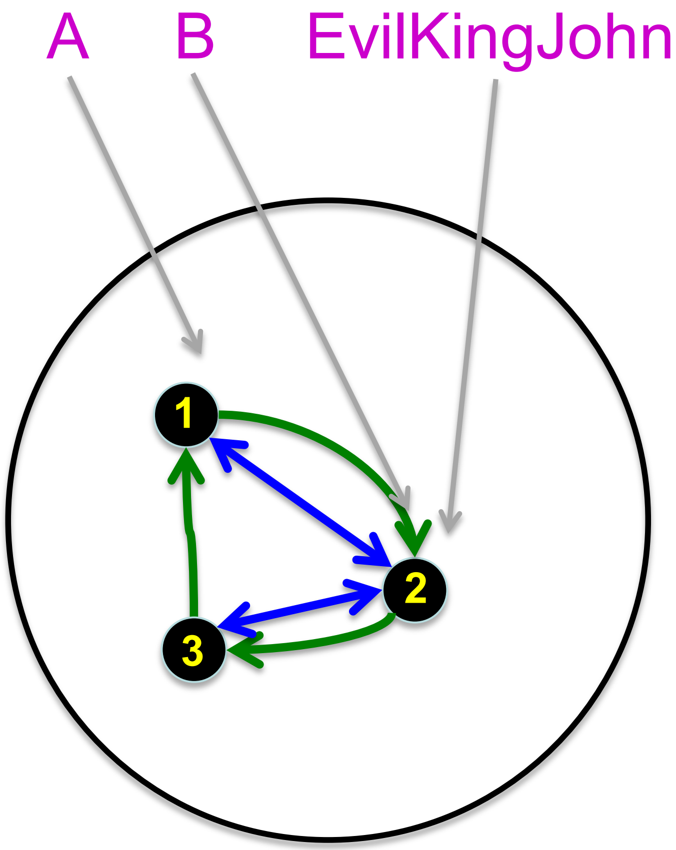
Syntax and semantics: Complex sentences

- Sentences with logical connectives
  $\neg\alpha$, $\alpha \wedge \beta$, $\alpha \vee \beta$, $\alpha \Rightarrow \beta$, $\alpha \Leftrightarrow \beta$
- Sentences with universal or existential quantifiers, e.g.,
  - $\forall$x Knows(x,BFF(x))
    - True in world w iff true in ***all extensions*** of w where x refers to an object in w
      - x -> **1**: Knows(**1**,BFF(**1**)) -> Knows(**1**,**2**) -> **T**
      - x -> **2**: Knows(**2**,BFF(**2**)) -> Knows(**2**,**3**) -> **T**
      - x -> **3**: Knows(**3**,BFF(**3**)) -> Knows(**3**,**1**) -> **F**



Syntax and semantics: Complex sentences

- Sentences with logical connectives
  $\neg\alpha$, $\alpha \wedge \beta$, $\alpha \vee \beta$, $\alpha \Rightarrow \beta$, $\alpha \Leftrightarrow \beta$
- Sentences with universal or existential quantifiers, e.g.,
  - $\exists$x Knows(x,BFF(x))
    - True in world w iff true in ***some extension*** of w where x refers to an object in w
      - x -> **1**: Knows(**1**,BFF(**1**)) -> Knows(**1**,**2**) -> **T**
      - x -> **2**: Knows(**2**,BFF(**2**)) -> Knows(**2**,**3**) -> **T**
      - x -> **3**: Knows(**3**,BFF(**3**)) -> Knows(**3**,**1**) -> **F**



Fun with sentences

- Everyone knows President Obama

- There is someone that everyone knows

- Everyone knows someone

More fun with sentences

- Any two people of the same nationality speak a common language

Inference in FOL

- Entailment is defined exactly as for PL:
  - $\alpha \models \beta$ ("α entails β" or "β follows from α") iff in every world where α is true, β is also true
  - E.g., $\forall x\ Knows(x,Obama)$ entails $\exists y \forall x\ Knows(x,y)$
- If asked "Do you know what time it is?", it's rude to say "Yes"
- Similarly, given an existentially quantified query, it's polite to provide an answer in the form of a ***substitution*** (or ***binding***) for the variable(s):
  - KB = $\forall x\ Knows(x,Obama)$
  - Query = $\exists y \forall x\ Knows(x,y)$
  - Answer = Yes, $\{y/Obama\}$
- Applying the substitution should produce a sentence that is entailed by KB

Inference in FOL: Propositionalization

- Convert (KB $\wedge \neg\alpha$) to PL, use a PL SAT solver to check (un)satisfiability
  - Trick: replace variables with ground terms, convert atomic sentences to symbols
    - $\forall$x Knows(x,Obama) and Democrat(Feinstein)
      - Knows(Obama,Obama) and Knows(Feinstein,Obama) and Democrat(Feinstein)
      - K_O_O $\wedge$ K_F_O $\wedge$ D_F
    - and $\forall$x Knows(Mother(x),x)
      - Knows(Obama,Obama) and Knows(Mother(Obama),Obama) and Knows(Mother(Mother(Obama)),Obama) …….
  - Real trick: for k = 1 to infinity, use terms of function nesting depth k
    - If entailed, will find a contradiction for some finite k; if not, may continue for ever; ***semidecidable***

Inference in FOL: Lifted inference

- Apply inference rules directly to first-order sentences, e.g.,
  - KB = Person(Socrates), $\forall$x Person(x) $\Rightarrow$ Mortal(x)
  - conclude Mortal(Socrates)
  - The general rule is a version of Modus Ponens:
    - Given $\alpha[x] \Rightarrow \beta[x]$ and α', where $\alpha'\sigma = \alpha[x]\sigma$ for some substitution σ conclude $\beta[x]\sigma$
      - σ is {x/Socrates}
    - Given Knows(x,Obama) and Knows(y,z) $\Rightarrow$ Likes(y,z)
      - σ is {y/x, z/Obama}, conclude Likes(x,Obama)
- Examples: Prolog (backward chaining), Datalog (forward chaining), production rule systems (forward chaining), resolution theorem provers

Summary, pointers

- FOL is a very expressive formal language
- Many domains of common-sense and technical knowledge can be written in FOL (see AIMA Ch. 12)
  - circuits, software, planning, law, network and security protocols, product descriptions, ecommerce transactions, geographical information systems, Google Knowledge Graph, Semantic Web, etc.
- Inference is semidecidable in general; many problems are efficiently solvable in practice
- Inference technology for logic programming is especially efficient (see AIMA Ch. 9)