First Order Logic

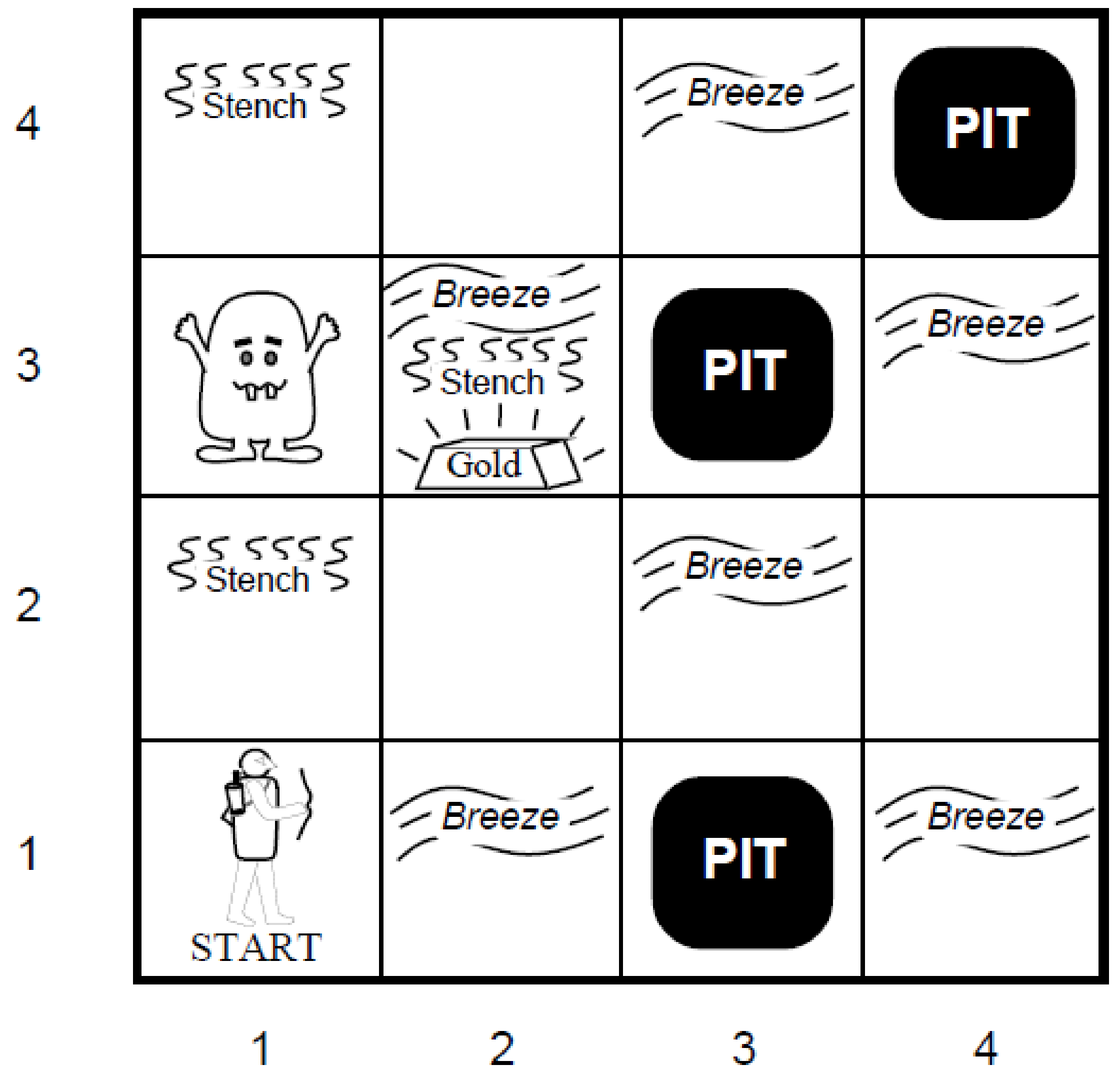
Dr. Asaad Sabah Hadi

Knowledge Representation & Reasoning

❑ Introduction

- ✓ Propositional logic is declarative
- ✓ Propositional logic is compositional: meaning of B1,1 ∧ P1,2 is derived from meaning of B1,1 and of P1,2
- ✓ Meaning in propositional logic is context-independent unlike natural language, where meaning depends on context
- ✓ Propositional logic has limited expressive power unlike natural language

e.g., cannot say "pits cause breezes in adjacent squares“
(except by writing one sentence for each square)


4
3
2
1
Stench
Breeze
PIT
Breeze
Stench
Gold
PIT
Breeze
Stench
Breeze
START
Breeze
PIT
Breeze
1
2
3
4

Knowledge Representation & Reasoning

- From propositional logic (PL) to First order logic (FOL)

✓ Examples of things we can say:

All men are mortal:

- $\forall x\ Man(x) \Rightarrow Mortal(x)$

Everybody loves somebody

- $\forall x\ \exists y\ Loves(x, y)$

The meaning of the word “above”

- $\forall x\ \forall y\ above(x,y) \Leftrightarrow (on(x,y) \vee \exists z\ (on(x,z) \wedge above(z,y))$

First Order Logic

- Whereas propositional logic assumes the world contains facts, first-order logic (like natural language) assumes the world contains:

- Objects: people, houses, numbers, colors, …
- Relations: red, round, prime, brother of, bigger than, part of, …
- Functions: Sqrt, Plus, …

- **Can express the following:**
  - Squares neighboring the Wumpus are smelly;
  - Squares neighboring a pit are breezy.

Syntax Order Logic

User defines these primitives:

1. **Constant symbols** (i.e., the "individuals" in the world) e.g., Mary, 3
2. **Function symbols** (mapping individuals to individuals) e.g., father-of(Mary) = John, colorof(Sky) = Blue
3. **Predicate/relation symbols** (mapping from individuals to truth values) e.g., greater(5,3), green(apple), color(apple, Green)

❑ Syntax Order Logic

FOL supplies these primitives:

1. **Variable symbols**. e.g., x,y
2. **Connectives.** Same as in PL: ⇔,∧,∨, ⇒
3. **Equality** =
4. **Quantifiers**: Universal (∀) and Existential (∃)

A legitimate expression of predicate calculus is called a well-formed formula (wff) or, simply, a sentence.

Knowledge Representation & Reasoning

Syntax Order Logic

Quantifiers: Universal (∀) and Existential (∃)

Allow us to express properties of collections of objects instead of enumerating objects by name

Universal: “for all”:

∀<variables> <sentence>

Existential: “there exists”

∃<variables> <sentence>

Knowledge Representation & Reasoning

- Syntax Order Logic: **Constant Symbols**

- A symbol, e.g. Wumpus, Ali.

- Each constant symbol names exactly one object in a **universe of discourse**, but:
  - not all objects have symbol names;
  - some objects have several symbol names.

- Usually denoted with upper-case first letter.

Knowledge Representation & Reasoning

- Syntax Order Logic: **Variables**
- Used to represent objects or properties of objects without explicitly naming the object.
- Usually lower case.
- For example:
  - x
  - father
  - square

.

Knowledge Representation & Reasoning

Syntax Order Logic: **Relation (Predicate) Symbols**

- A predicate symbol is used to represent a relation in a universe of discourse.
- The sentence

  Relation(Term1, Term2,…)

  is either TRUE or FALSE depending on whether Relation holds of Term1, Term2,…

- To write "Malek wrote Muata" in a universe of discourse of names and written works:

Wrote(Malek, Muata)

This sentence is true in the intended interpretation.

- Another example:

  Instructor (CAP492, James)

Syntax Order Logic: Function symbols

- Functions talk about the binary relation of pairs of objects.

- For example, the Father relation might represent all pairs of persons in father-daughter or father-son relationships:
  - Father(Ali) Refers to the father of Ali
  - Father(x) Refers to the father of variable x

Knowledge Representation & Reasoning

Syntax Order Logic: properties of quantifiers

- $\forall x \forall y$ is the same as $\forall y \forall x$
- $\exists x \exists y$ is the same as $\exists y \exists x$
- $\exists x \forall y$ is not the same as $\forall y \exists x$:
- $\exists x \forall y$ Loves(x,y)
- “There is a person who loves everyone in the world”
- $\forall y \exists x$ Loves(x,y)
- “Everyone in the world is loved by at least one person”
- Quantifier duality: each can be expressed using the other
- $\forall x$ Likes(x,IceCream) $\equiv \neg \exists x \neg$Likes(x,IceCream)
- $\exists x$ Likes(x,Broccoli) $\equiv \neg \forall x \neg$Likes(x,Broccoli)

Syntax Order Logic: Atomic sentence

Atomic sentence = ***predicate*** ($term_1$,...,$term_n$)

or $term_1 = term_2$

Term = ***function*** ($term_1$,...,$term_n$)

or ***constant*** or ***variable***

Example terms:

Brother(Ali , Mohamed)

Greater(Length(x), Length(y))

Syntax Order Logic: Complex sentence

Complex sentences are made from atomic sentences using connectives and by applying quantifiers.

<u>Examples:</u>

- *Sibling(Ali,Mohamed)* $\Rightarrow$ *Sibling(Mohamed,Ali)*
- greater(1,2) $\vee$ less-or-equal(1,2)
- $\forall$ x,y Sibling(x,y) $\Rightarrow$ Sibling(y,x)

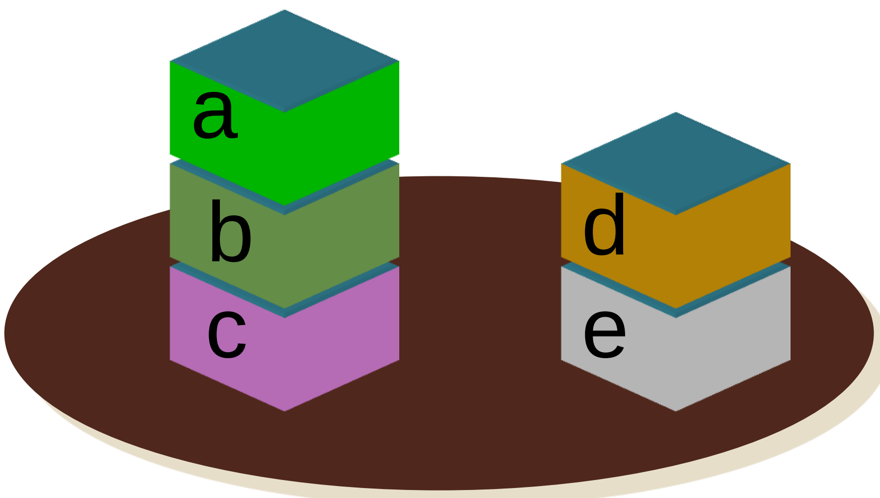
- **While constant symbols, variables and connectives are like propositional logic, "What are functions and predicates?"**

The language of logic is based on set theory:

- Sets;
- Relations;
- Functions.

Knowledge Representation & Reasoning

SETS: The set of objects defines a "Universe of Discourse." [Objects are represented by Constant Symbols.]



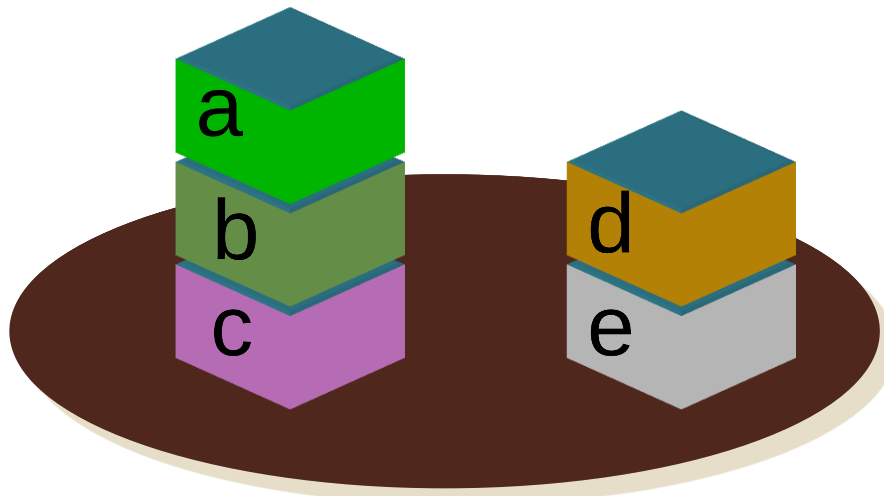
Interpretation:

A → a
B → b
C → c
D → d
E → e

For example, in this blocks world, the universe of discourse is {a,b,c,d,e}.

Knowledge Representation & Reasoning

- **RELATIONS:** Def: A **binary relation** is a set of ordered pairs
- Example: Consider set of blocks {a,b,c,d,e}



The "on" relation:
on = {<a,b>, <b,c>, <d,e>}.
The predicate On(A,B) can be **interpreted** as: <a,b> $\in$ on.
On(A,B) is TRUE, but On(A,C) and On(C,D) are FALSE in this interpretation.

- **FUNCTIONS:**

A function is a binary relation such that no two distinct members have the same first element.

In other words, if F is a function

$\langle x,y\rangle \in F$ and $\langle x,z\rangle \in F \quad \Rightarrow y=z$

If $\langle x,y\rangle \in F$:

x is an argument of F ;

y is the value of F at x ;

y is the image of x under F.

F(x) designates the object y such that y=F(x).

Knowledge Representation & Reasoning

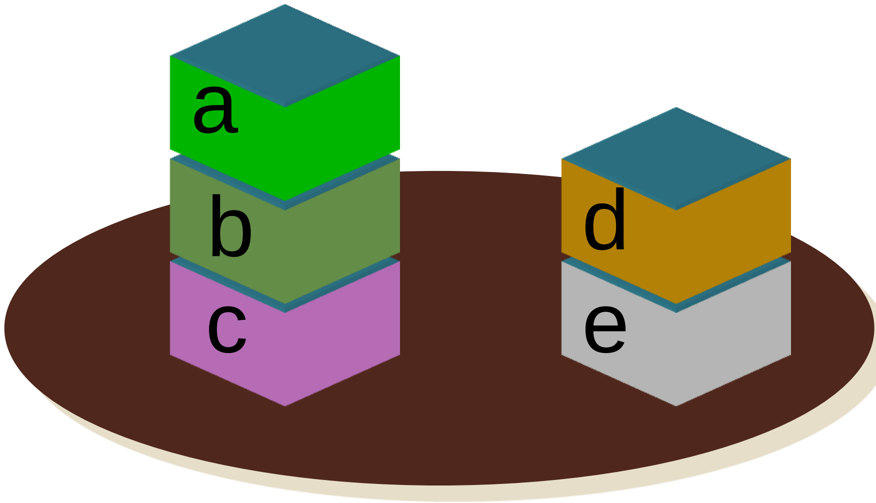
- **FUNCTIONS:**

hat = {<c,b>, <b,a>,<e,d>}

hat (c) = b

hat(b) = a

hat(d) is not defined.



Hat(e) can be interpreted as d

Using FOL

On(A,B)

On(B,C)

On(D,E)

On(A,Hat(C))

Knowledge Representation & Reasoning

Syntax of First Order Logic

Sentence → ***Atomic Sentence***
| (***sentence connective Sentence***)
| ***Quantifier variable,… Sentence***
| ¬ ***Sentence***

Atomic Sentence → ***Predicate*** (***Term,…***) | ***Term=Term***
Term → ***Function***(***Term,…***) | ***Constant*** | ***variable***

Connective → ⇔ | ∧ | ∨ | ⇒
Quantifier → ∀ | ∃
Constant → A | X_1…
Variable → a | x | s | …
Predicate → Before | hascolor | ….
Function → Mother | Leftleg |…

Inference in First Order Logic

Inference in FOL can be performed by:

- ✓ Reducing first-order inference to propositional inference
- ✓ Unification
- ✓ Generalized Modus Ponens
- ✓ Resolution
- ✓ Forward chaining
- ✓ Backward chaining

Inference in First Order Logic

- **From FOL to PL**

First order inference can be done by converting the knowledge base to PL and using propositional inference.

Two questions??

How to convert universal quantifiers?

Replace variable by ground term.

How to convert existential quantifiers?

Skolemization.

Inference in First Order Logic

Substitution

Given a sentence α and binding list σ, the result of applying the **substitution** σ to α is denoted by Subst(σ, α).

Example:

σ = {x/Ali, y/Fatima} α = Likes(x,y)

Subst**(**{x/Sam, y/Pam}, Likes(x,y)**)** = Likes(Sam, Pam)

Inference in First Order Logic

Universal instantiation (UI)

Every instantiation of a universally quantified sentence is entailed by it:

$$\frac{\forall v\ \alpha}{\text{Subst}(\{v/g\}, \alpha)}$$

for any variable v and ground term g

e.g., $\forall x\ \text{King}(x) \land \text{Greedy}(x) \Rightarrow \text{Evil}(x)$ yields:

$\text{King}(\text{John}) \land \text{Greedy}(\text{John}) \Rightarrow \text{Evil}(\text{John})$

$\text{King}(\text{Richard}) \land \text{Greedy}(\text{Richard}) \Rightarrow \text{Evil}(\text{Richard})$

$\text{King}(\text{Father}(\text{John})) \land \text{Greedy}(\text{Father}(\text{John})) \Rightarrow \text{Evil}(\text{Father}(\text{John}))$

Inference in First Order Logic

Existential instantiation (EI)

For any sentence α, variable v, and constant symbol k that does not appear elsewhere in the knowledge base:

$$\frac{\exists v\ \alpha}{\mathrm{Subst}(\{v/k\}, \alpha)}$$

e.g., $\exists x\ Crown(x) \land OnHead(x, John)$ yields:

$Crown(C1) \land OnHead(C1, John)$

provided *C1* is a new constant symbol, called a Skolem constant

Inference in First Order Logic

Reduction to propositional inference

Suppose the KB contains just the following:

$\forall x \; King(x) \land Greedy(x) \Rightarrow Evil(x)$

$King(John)$

$Greedy(John)$

$Brother(Richard, John)$

- Instantiating the universal sentence in all possible ways, we have:

$King(John) \land Greedy(John) \Rightarrow Evil(John)$

$King(Richard) \land Greedy(Richard) \Rightarrow Evil(Richard)$

- The new KB is propositionalized

Inference in First Order Logic

- Instead of translating the knowledge base to PL, we can make the inference rules work in FOL.

For example, given

$\forall$ x King(x) $\wedge$ Greedy(x) $\Rightarrow$ Evil(x)

King(John)

$\forall$ y Greedy(y)

It is intuitively clear that we can substitute {x/John,y/John} and obtain that Evil(John)

Knowledge Representation & Reasoning

Inference in First Order Logic

2. Unification

- We can make the inference if we can find a substitution such that *King(x)* and *Greedy(x)* match *King(John)* and *Greedy(y)*

  {x/John,y/John} works

- Unify(α ,β) = θ if Subst(θ, α) = Subst(θ, β)

α	β	Subst
Knows(John,x)	Knows(John,Jane)	{x/Jane}
Knows(John,x)	Knows(y,OJ)	{x/OJ,y/John}
Knows(John,x)	Knows(y,Mother(y))	{y/John,x/Mother(John)}}
Knows(John,x)	Knows(x,OJ)	{fail}

Knowledge Representation & Reasoning

Inference in First Order Logic

3. Generalized Modus Ponens (GMP)

- Suppose that Subst(θ, *pi'*) = Subst(θ, *pi*) *for all i then:*

$$\frac{p1', p2', \ldots, pn', (p1 \land p2 \land \ldots \land pn \Rightarrow q)}{\text{Subst}(\theta, q)}$$

- p1' is *King*(*John*) p1 is *King*(*x*)
- p2' is *Greedy*(*y*) p2 is *Greedy*(*x*)
- θ is {x/John,y/John} q is *Evil*(*x*)

- Subst(θ, q) is *Evil*(*John*)

All variables assumed universally quantified.

Inference in First Order Logic

3. Resolution

Full first-order version:

$$\frac{l_1 \vee \cdots \vee l_k, \qquad m_1 \vee \cdots \vee m_n}{Subst(\theta, l_1 \vee \cdots \vee l_{i-1} \vee l_{i+1} \vee \cdots \vee l_k \vee m_1 \vee \cdots \vee m_{j-1} \vee m_{j+1} \vee \cdots \vee m_n)}$$

where $\theta = Unify(l_i, \neg m_j)$

$$\frac{\neg Rich(x) \vee Unhappy(x), \quad Rich(Ken)}{Unhappy(Ken)}$$

with $\theta = \{x/Ken\}$

Apply resolution steps to $CNF(KB \wedge \neg\alpha)$; complete for FOL

Inference in First Order Logic

4. Forward chaining

When a new fact P is added to the KB:

For each rule such that P unifies with a premise,

if the other premises are known

then add the conclusion to the KB and continue chaining.

Forward chaining is data-driven,

e.g., inferring properties and categories from percepts.

Knowledge Representation & Reasoning

Inference in First Order Logic

4. Forward chaining example.

- Rules

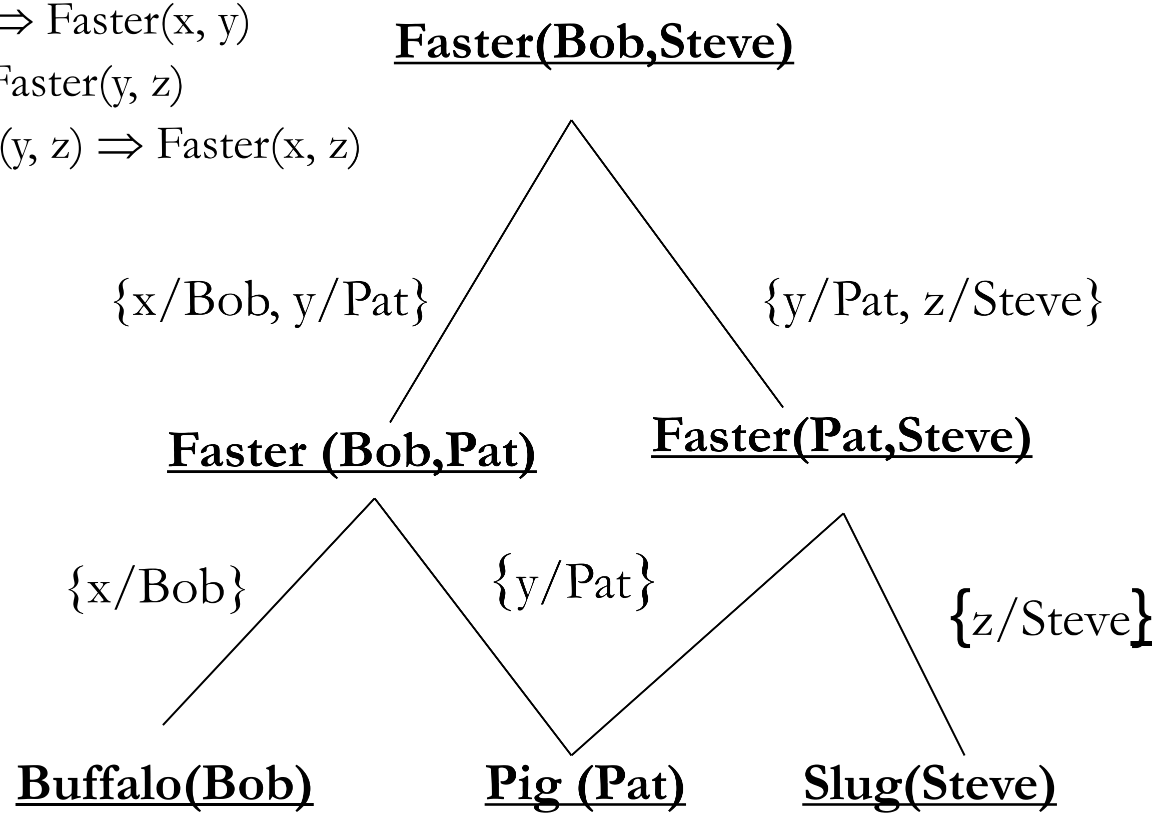
1. Buffalo(x) $\wedge$Pig(y) $\Rightarrow$ Faster(x, y)
2. Pig(y) $\wedge$Slug(z) $\Rightarrow$ Faster(y, z)
3. Faster(x, y) $\wedge$Faster(y, z) $\Rightarrow$ Faster(x, z)

Facts

1. Buffalo(Bob)
2. Pig(Pat)
3. Slug(Steve)

New facts

4. Faster(Bob,Pat)
5. Faster(Pat, Steve)
6. Faster (Bob, Steve)



Knowledge Representation & Reasoning

Inference in First Order Logic

4. Backward chaining

Backward chaining starts with a hypothesis and work backwards , according to the rules in the knowledge base until reaching confirmed findings or facts.

Pig(y)∧Slug(z) ⇒ Faster (y, z)

Slimy(a)∧ Creeps(a) ⇒Slug(a)

Pig(Pat)

Slimy(Steve)

Creeps(Steve)