



人工知能と 意味の形式的理論

The meaning of a complex expression is determined by the meanings of its parts and rules used for combining them.

--- Gottlob Frege

I reject the contention that an important theoretical difference exists between formal and natural languages.

--- Richard Montague

To conceptually relate any two things, it is necessary that they belong to a common category. the clearest exposition of the whole matter may involve presenting the two things themselves as citizens of that third category.

Thus, if we construe theories as categories, models are functors!

--- F. W. Lawvere

はじめに

- 人工知能技術にとって、自然言語の意味の理解は、重要な課題である。小論は、自然言語の意味を形式的に把握しようという試みを概観したものである。
- 第一部では、まず、現在の主要な三つの自然言語処理技術の現状を紹介し、あわせて、言語の意味理解にフォーカスして、様々な取り組みを取り上げた。
- こうした技術を評価する上で、筆者の取っている基本的な視点は、次のようなものである。

基本的な観察

文と意味の「構成性(compositionality)」

- 文は文法に従って語から構成される
- 文の意味は、語の意味に依存する
 - 文法的に正しいが意味がない文もある
Colorless green ideas sleep furiously.
- 文の意味は、文の文法的構成に依存する
 - 二つの意味を持つ、あいまいな文
I saw a man with a telescope.

意味の「同一性」 / 意味の共通表現の存在

- 言語が異なっても、「同じ」意味を持つ語が存在する。
- 言語が異なっても、「同じ」意味を持つ文が存在する。

はじめに

- 残念ながら、文が語から文法に基づいて構成されることは、現在主流の自然言語処理技術では、ほとんど考慮されていない。文法性の認識がないのでは、文の意味の構成性の認識を持つことは難しい。
- ただ、文の意味の構成性の認識なしにでも、意味については考えることができる。一つには文を構成する「語の意味」、もう一つには「意味の同一性」に基づく「意味の共通表現」の模索である。第一部の後半では、これらの取り組みを取り上げた。
- 「語の意味」の表現では、その客観性・共通性を「実在」の関係に基礎をもつOntology、語の利用の頻度の統計的分析に帰着させるWord2Vec的「分散表現」、辞書項目に諸特徴を枚挙するスタイル、conceptual spacesを構成するアプローチ等多様な試みが行われている。

はじめに

- 「文の意味」の表現については、論理式（あるいは、ラムダ式）による表現と多次元ベクトルによる分散表現の二つがある。後者は、実装者にはそういうものとしては、あまり自覚されていないようにみえるのだが。
- 機械翻訳技術の成功は、二つの言語の意味の「共通表現」を多次元ベクトルによる分散表現として抽出しているところにあると筆者は考えている。もっとも、語の意味も、文の文法性も、このアプローチでは、直接には考慮されていない。
- 筆者は、論理式による表現が「好み」なのだが、文から論理式への還元は、文法に応じて様々な流儀がある。この点で最もスマートなのは、CCG (Combinatory Categorical Grammars) のやり方である。このアプローチは、決して最新の技術ではないのだが、第三部で取り上げることにする。

はじめに

- 筆者の基本的な問題意識は、文の文法に基づく構成性と文の意味の構成性とを結びつけることである。
- 第三部で紹介するCCG (Combinatory Categorical Grammars)と、DisCoCat(DIStributional COmpositional CATegorical)は、こうした要請に応える言語理論だと、筆者は考えている。
- 文と意味の構成性(Compositionality)は、言語理論の形式性の基礎である。それは、数学的には、文法と意味はカテゴリーとして記述できるということに、ほぼ等しいのだ。
- 第二部は、第三部の準備として、LambekのPregroup Grammarの紹介とLawvereのFunctor Semantic の初等的な解説にあてた。

Agenda

Part I

自然言語処理技術と意味理解の試み

Part II

構成的意味理解への予備的アプローチ

Part III

意味の形式的理論の現在

Appendix

Quantum Semantics

Part I 自然言語処理技術と意味理解の試み

Agenda

- 現在の自然言語処理技術と三つの主要なモデル
 - Intentモデル
 - Entityモデル
 - 機械翻訳モデル
- 意味の表現をめぐる様々な試み I -- Deep Learning --
 - Word2Vec
 - Scene Graph
 - Image Sentence Mapping
- 意味の表現をめぐる様々な試み II -- 意味の論理的表現 --
 - Minimal Recursion Semantics
 - Deep parsing in Watson

Part II 構成的意味理解への予備的アプローチ

Agenda

- Minimalist Programと Categorical Grammar
- Lambek vs Lambek
 - Categorical GrammarとPregroup Grammar
 - The mathematics of sentence structure
 - From word to sentence
- 理論とモデル
 - 意味の二項性
 - 「ルール」の世界と「たとえ」の世界
 - 語の意味のモデル – 素朴な対応理論
 - 理論とモデル – 対応の形式化
 - 理論とモデル 2 – 数学でのモデルの利用
 - Category論とFunctor Semantic

Part III 意味の形式的理論の現在

Agenda

- CCG (Combinatory Categorical Grammars)での Semantic Parsing
 - Semantic Parsing with Combinatory Categorical Grammars
- DisCoCat (DIStributIonal Compositional CATegorical) 意味モデル
 - WHAT IS APPLIED CATEGORY THEORY ?
 - Compositionality in Semantic Spaces
 - Mathematical Foundations for a Compositional Distributional Model of Meaning

Appendix Quantum Semantics

- A quantum teleportation inspired algorithm produces sentence meaning from word meaning and grammatical structure
- Open System Categorical Quantum Semantics in Natural Language Processing

Part I

自然言語処理技術と 意味理解の試み



Part I 自然言語処理技術と意味理解の試み

Agenda

- 現在の自然言語処理技術と三つの主要なモデル
 - Intentモデル
 - Entityモデル
 - 機械翻訳モデル
- 意味の表現をめぐる様々な試み I -- Deep Learning --
 - Word2Vec
 - Scene Graph
 - Image Sentence Mapping
- 意味の表現をめぐる様々な試み II -- 意味の論理的表現 --
 - Minimal Recursion Semantics
 - Deep parsing in Watson

現在の自然言語処理技術と 三つの主要なモデル

Intentモデル
Entityモデル
機械翻訳モデル

Alexa Intentモデル

Intentモデル

- Alexaなどのパーソナル・アシスタントに利用されているモデル。
- 「チューリング・テスト」やサールの「中国人の部屋」と同様の操作主義的な「知性」の解釈
- 人間の側の発話の意図を機械が把握し、その意図に応じて正しく反応するのが、機械にとっての「意味理解」。
- Syntax-Semantics-Pragmaticsの階層でのPragmaticな（「語用論」的）意味の理解。具体的には、
 - 命令文 — 命令された動作を行えばいい
 - 疑問文 — 答えを与えればいい —> 検索を実行
- 実装が容易。ある種の「型システム」を導入している。（後述）

Alexa Intentモデルの実装

- 人間の側の発話の意図を機械が把握し、その意図に応じて正しく反応するのが、「意味理解」
- 基本的には、人間の側の多様な発話を、次のような手段で制限しながら、パターンマッチングで音声コマンドとして把握する。
 - **Skill**: 発話のドメインを制限する。
 - **Intent** : トピック。発話の意図を大きく指定する。
 - **Slot**: Intentの引数。Intentが関連する意味を具体化する役割を持つ。マッチした値は、機械の側の発話にも利用できる。
 - **Slot type**: Date, Location, ... 等の型とそのインスタンスのサポート。人間の側の発話とSlot typeのインスタンスとの自動マッチング。Schemaのサポート。
 - **Intent signature**: あるIntentのもとでの文例のパターンを提供する。ある種の文法構造の記述。

Slot Type

Numbers, Dates,
and Times

Slot Type	Short Description	Supported Languages
AMAZON.DATE	Converts words that indicate <i>dates</i> ("today", "tomorrow", or "july") into a date format (such as "2015-07-00T9").	English (UK) English (US) German
AMAZON.DURATION	Converts words that indicate <i>durations</i> ("five minutes") into a numeric duration ("PT5M").	English (UK) English (US) German
AMAZON.FOUR_DIGIT_NUMBER	Converts words that represent <i>four-digit numbers</i> ("six oh four five" into digits ("6045").	English (UK) English (US) German
AMAZON.NUMBER	Converts numeric words ("five") into digits (such as "5").	English (UK) English (US) German
AMAZON.TIME	Converts words that indicate time ("four in the morning", "two p m") into a time value ("04:00", "14:00").	English (UK) English (US) German

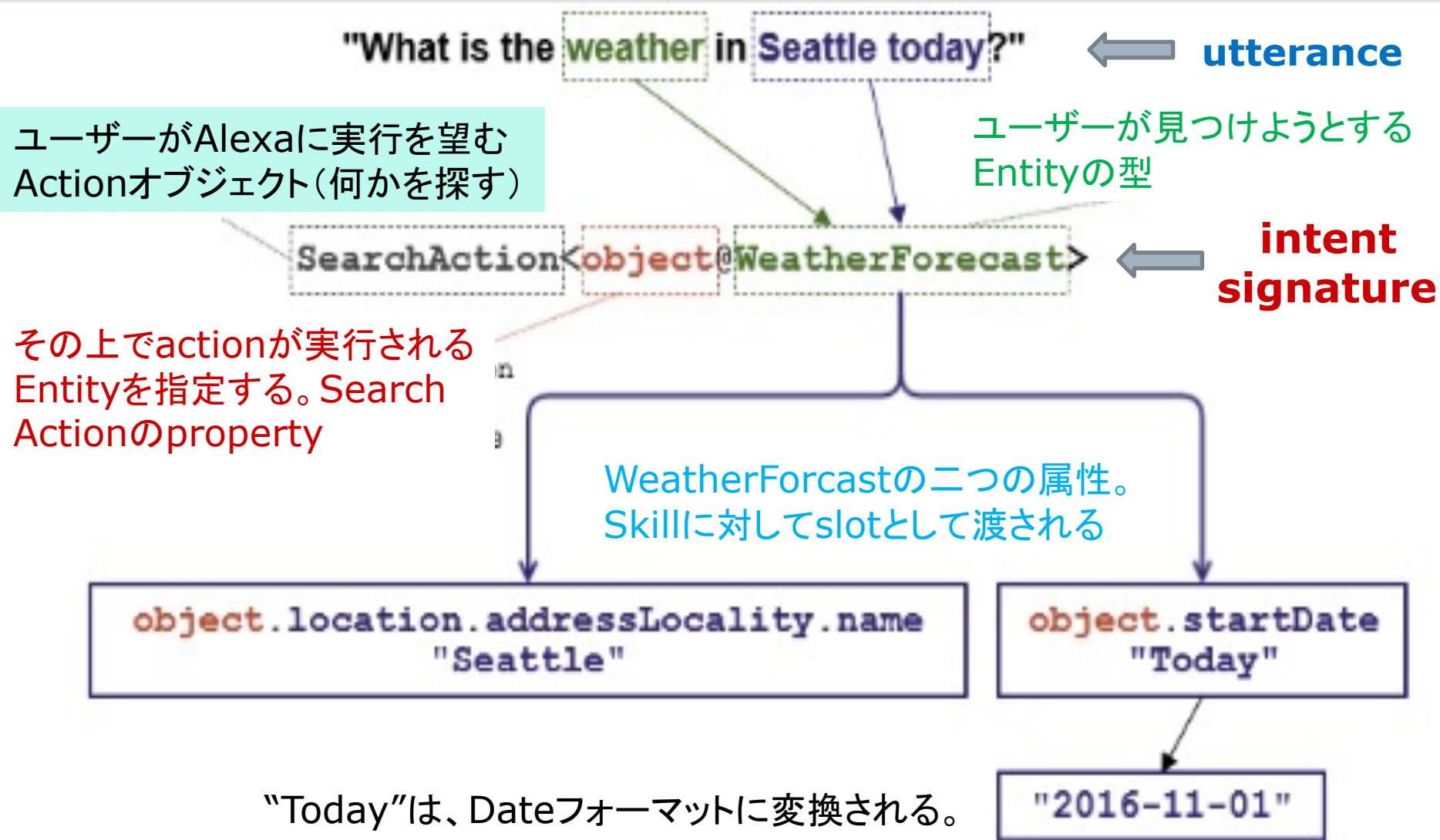
AMAZON.Actor
AMAZON.AdministrativeArea
AMAZON.AggregateRating
AMAZON.Airline
AMAZON.Airport
AMAZON.Animal
AMAZON.Artist
AMAZON.AT_CITY
AMAZON.AT_REGION
AMAZON.Athlete
AMAZON.Author
AMAZON.Book
AMAZON.BookSeries
AMAZON.BroadcastChannel
AMAZON.CivicStructure
AMAZON.Color
AMAZON.Comic
AMAZON.Corporation
AMAZON.Country
AMAZON.CreativeWorkType
AMAZON.DayOfWeek
AMAZON.DE_CITY

AMAZON.DE_FIRST_NAME
AMAZON.DE_REGION
AMAZON.Dessert
AMAZON.DeviceType
AMAZON.Director
AMAZON.Drink
AMAZON.EducationalOrganization
AMAZON.EUROPE_CITY
AMAZON.EventType
AMAZON.Festival
AMAZON.FictionalCharacter
AMAZON.FinancialService
AMAZON.Food
AMAZON.FoodEstablishment
AMAZON.Game
AMAZON.GB_CITY
AMAZON.GB_FIRST_NAME
AMAZON.GB_REGION
AMAZON.Genre
.....
.....
AMAZON.WrittenCreativeWorkType

Slot Type

List Type

AlexaのIntent Signature



Entityモデル

Entity モデル

- Google/MS等の大規模検索や知識表現/検索に利用されている、情報をEntityの構造に対応させる技術。
- あるものの「意味」は、対応するEntityが持つ「情報」として解釈される。
- Entityは、現実のものであれバーチャルなものであれ、具体的なものであれ抽象的なものであれ、概念としては「実在するもの」とみなされる。ある種の概念実在論。
- Entityの構造が表現するものは、実在的なものの構造の反映であるので(Ontology)、言語表現からは独立な、かつ、全ての言語で共通な情報を与えうる。
- 明確な「型システム」。
- 多くの実装は、Schema.orgのスキーマ定義に依存している。

Entity モデルの実装

- Entity は “is-a” と解釈される様々な「型」を持つ。
Maruyama “is-a” **Person**; Maruyama “is-a” **Fool**
その型は、“Thing”を最上位の型とする階層構造を持つ。
Thing > CreativeWork > Game > VideoGame
- Entity は、“has-a” と解釈される「属性(property)」を持つ。
この「属性」の値が、このEntityについての一つの情報である。
下位のEntityは、上位のEntityの属性とその値を継承する。
- それは、オブジェクト指向言語、例えば、Javaでの「クラス」と「フィールド」の関係に近いものである。ただし、Entityの型は多重継承を許す。

Entity Model と Action Type

- 先に見たように、Entity Modelは事物の客観的で存在論的な関係とそれが持つ情報を表現する。ただし、それは静的なものである。
- Schema.orgは2011年に発足するのだが、3年後の2014年、Action Typeを追加した。Schema blog: "Announcing Schema.org Actions" <https://goo.gl/eu2jmf>
- Thing > Action
- Thing > Intangible > Enumeration > ActionStatusType

<!-- John listened to Pink with Steve at Anna's apartment on his iPod. -->

```
<script type="application/ld+json">
{
  "@context": "http://schema.org",
  "@type": "ListenAction",
  "agent": {
    "@type": "Person",
    "name": "John"
  },
  "object": {
    "@type": "MusicGroup",
    "name": "Pink!"
  },
  "participant": {
    "@type": "Person",
    "name": "Steve"
  },
  "location": {
    "@type": "Residence",
    "name": "Ann's apartment"
  },
  "instrument": {
    "@type": "Product",
    "name": "iPod"
  }
}
```

Action Type を用いた、
文の意味の表現

Action Property

actionStatus

agent

endTime

instrument

error

location

object

participant

result

startTime

target

<https://schema.org/Action>

<!-- John created an exercise plan with Steve. -->

```
<script type="application/ld+json">
{
  "@context": "http://schema.org",
  "@type": "CreateAction",
  "agent": {
    "@type": "Person",
    "name": "John"
  },
  "result": {
    "@type": "ExercisePlan",
    "name": "John's weight loss plan"
  },
  "participant": {
    "@type": "Person",
    "name": "Steve"
  }
}
</script>
```

Action Type を用いた、
文の意味の表現

Action Property

actionStatus

agent

endTime

instrument

error

location

object

participant

result

startTime

target

<https://schema.org/Action>

-
- 文の文法性・構成性、意味の構成性への洞察を欠いている。
 - 手動での意味の「共通表現」の記述。それは、主に ActionType への恣意的なPropertyの追加によって担われている。
 - Schema 語彙の貧困。辞書のような網羅性を欠く。Word Netとの比較。
 - Folksonomy
-

機械翻訳モデル

機械翻訳モデル

- 機械翻訳に利用されるモデル。基本構成は、Encoder-Decoder + Attention。
- Intentモデル、Entityモデルとは異なって、Encoderが出力する**文の意味の表現モデルを内部に持つ**。それは、語の意味の表現モデルとしてのword2vecと同じく、多次元のベクトルで表現されている。**意味の分散モデル**。
- ニューラルネットは、文法を学習する能力を持つ。Attentionのメカニズムは、おそらく異なる言語間の文法の差異を吸収する役割を果たしている。
- 文の意味を表現する能力、文法を理解する能力を持つにも関わらず、その能力は、そうしたものとしては外部に取り出すことはできない。この技術を、翻訳以外の用途に転用するのは、難しい。

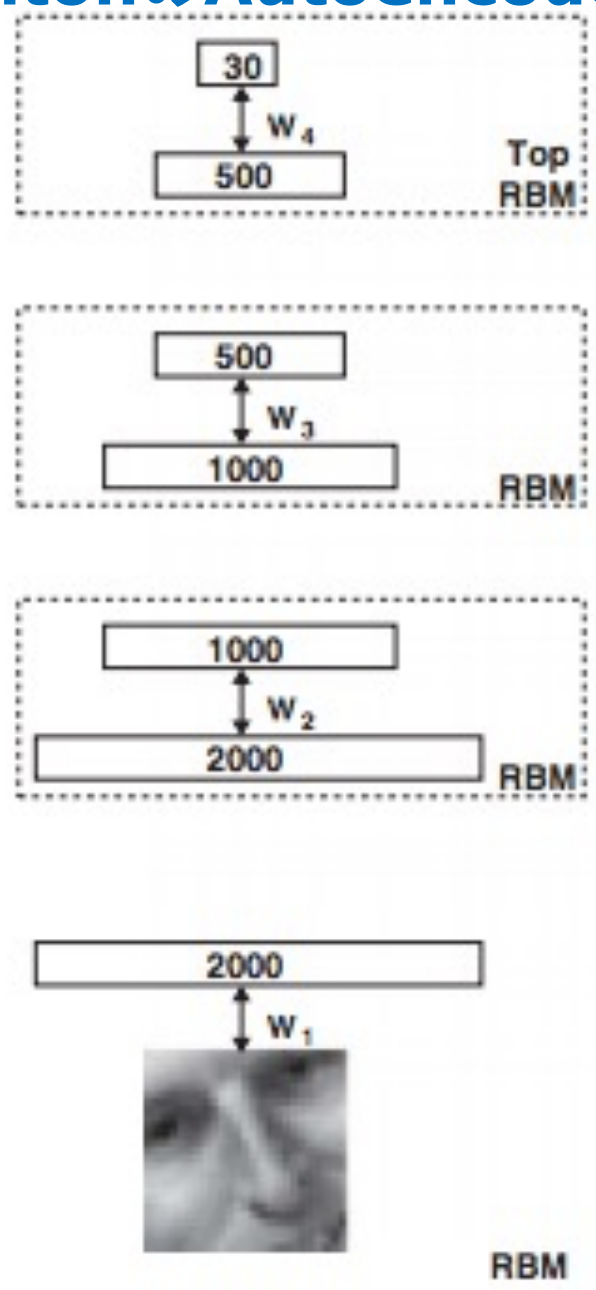
Encoder / Decoder

HintonのAutoencoder

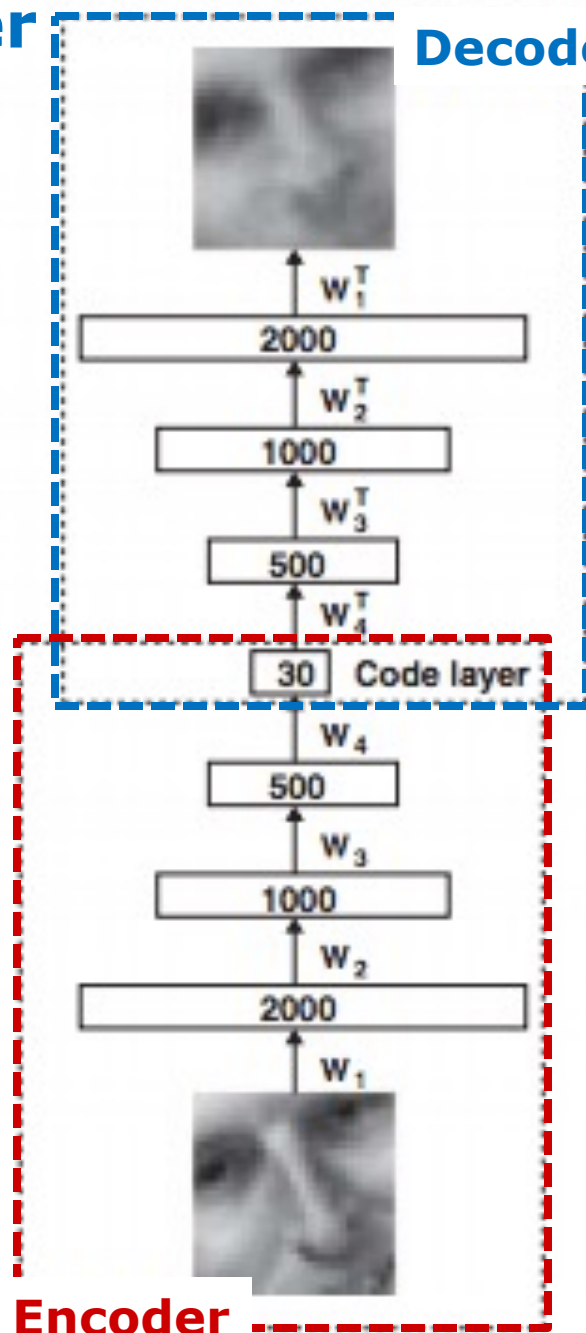
ニューラル・ネットワークの中で、Encoder-Decoderの枠組みを提案した基本的な論文は、Hintonの 2006年の **"Reducing the Dimensionality of Data with Neural Networks"** である。<https://goo.gl/Ki41L8>
Googleのニューラル機械翻訳は、この枠組みを利用している。

HintonのAutoencoder

Decoder

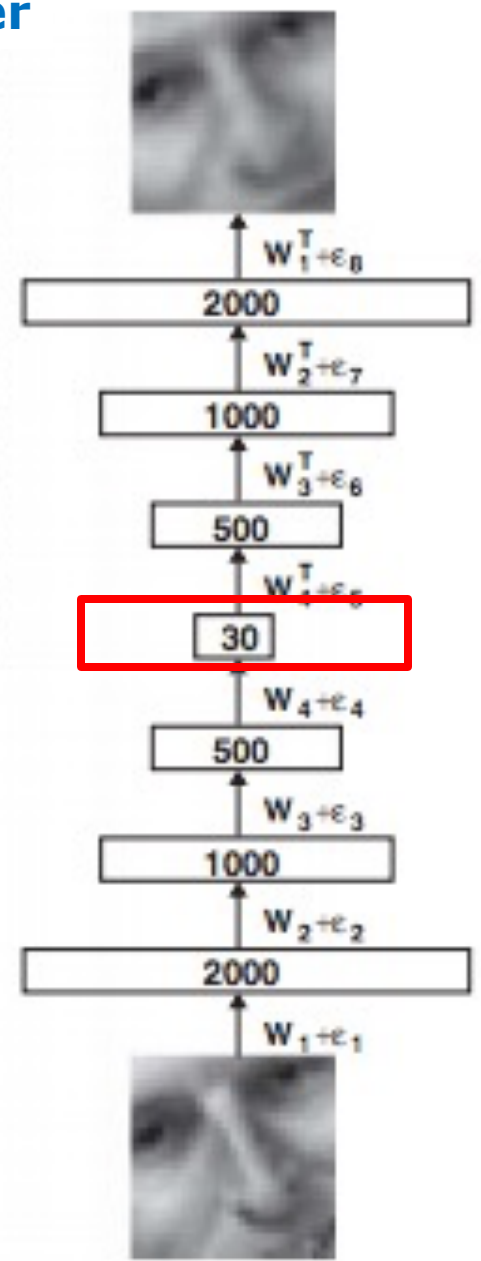


Pretraining



Encoder

Unrolling



Fine-tuning

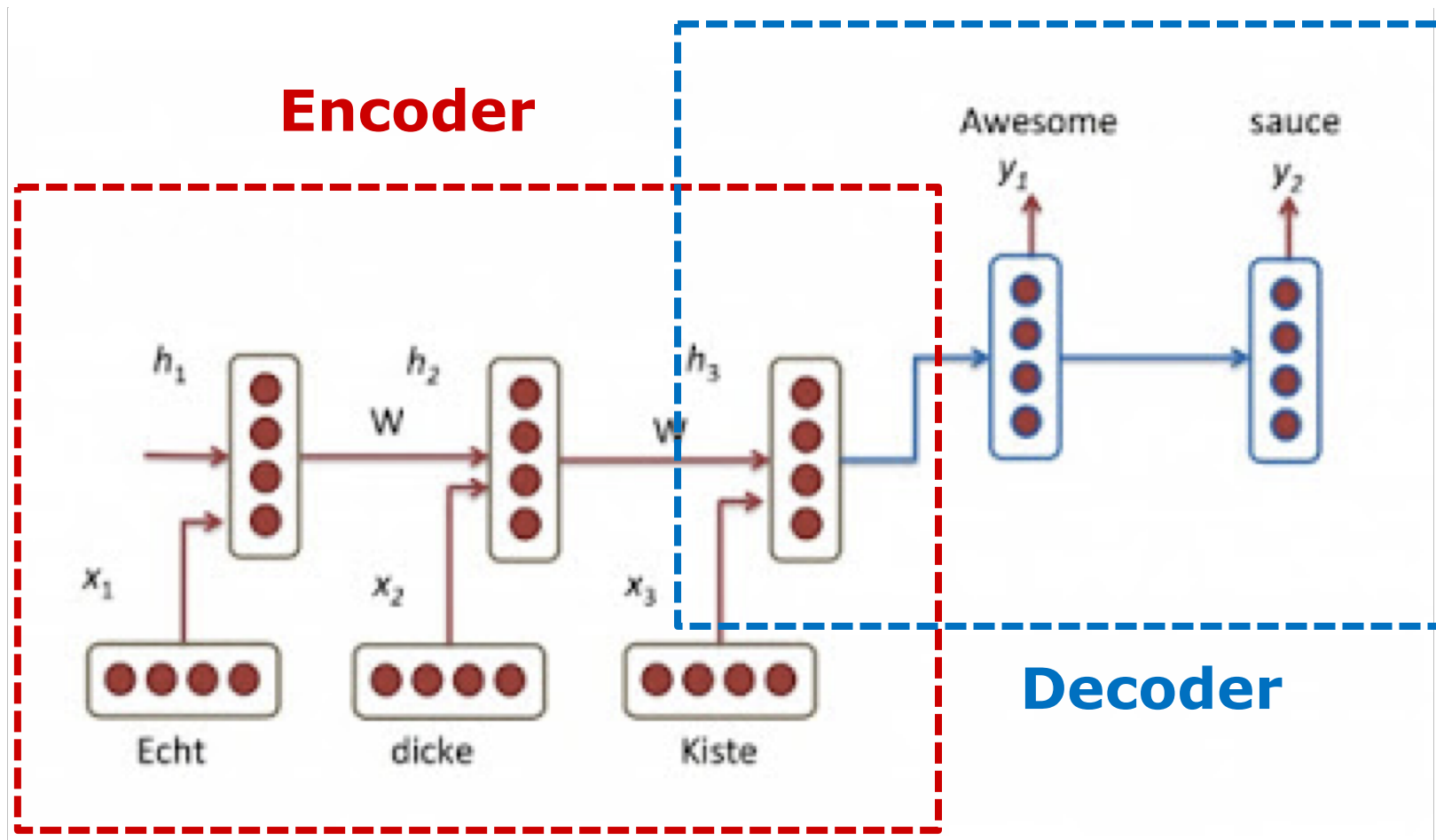
Semantic hashing(意味的ハッシング)

- Hintonは、こうしたAutoencoderの働きを、**Semantic hashing(意味的ハッシング)**と呼んでいる。SHA-1のようなハッシングでは、ハッシュ化されたデータから元のデータを復元することは不可能なのだが、Semantic hashingされたデータは、データの次元は低いものの、元の情報の中核部分を保持している。
- のちに、機械翻訳で利用される所以である。

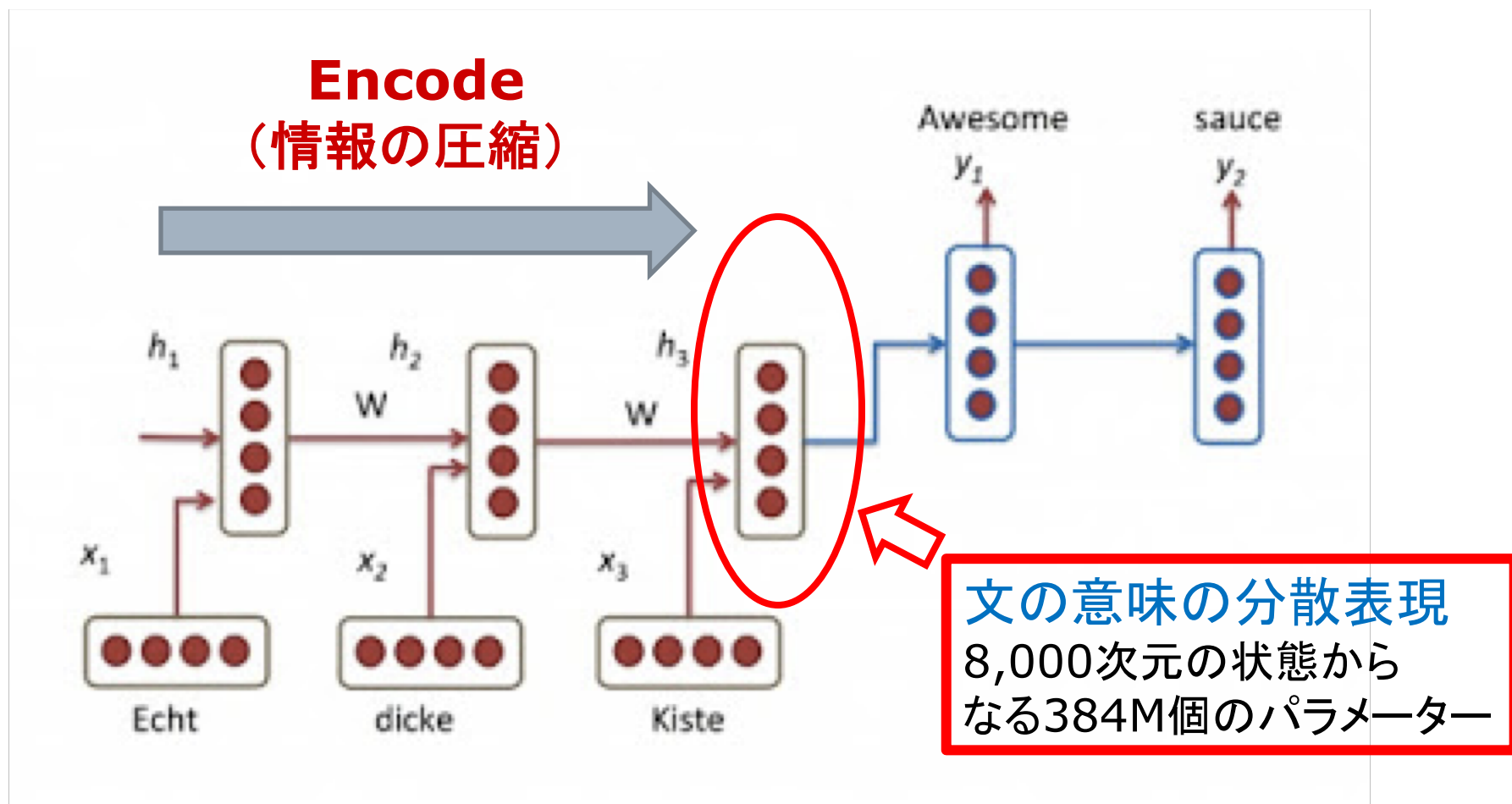
Encoder-Decoderの機械翻訳への応用

HintonのAutoencoderは、そのままの形では、機械翻訳に利用されることはなかったと思う。ただ、2014年に、Ilya Sutskever らは、RNN(ここでは多層のLSTM)の持つシーケンスをシーケンスに変換する能力が、機械翻訳に応用できるという論文 "**Sequence to Sequence Learning with Neural Networks**" <https://goo.gl/U3KtxJ> を発表する。

- 次の図(<https://goo.gl/JGckBP> から)は、こうしたメカニズムで、RNNが、独文の "Echt dicke Kiste" を英文の "Awesome sauce" に翻訳する様子を表している。(ここでは、文章の終わりを表す<EOS>は、省略されている)

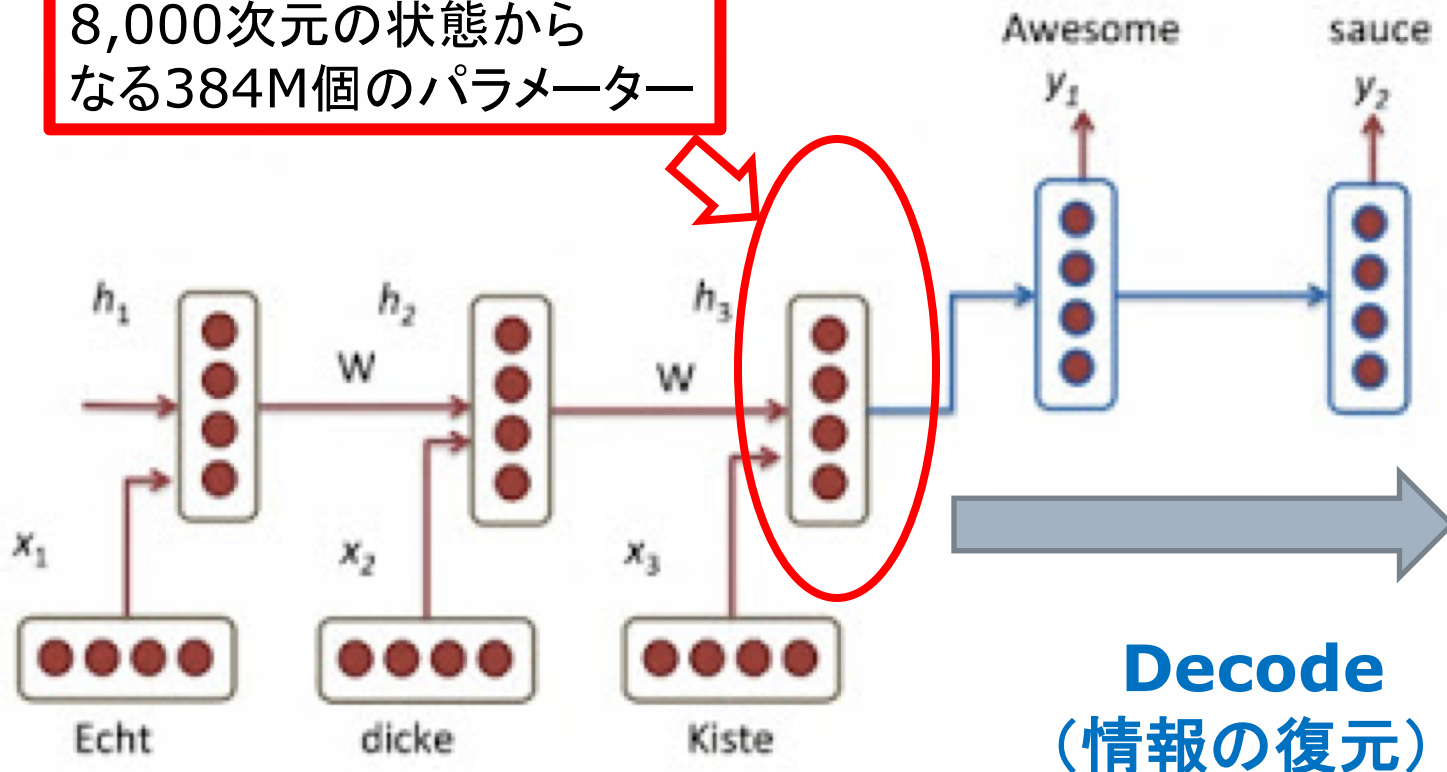


- ここでは、Encoder部が、文章の最後にとるRNNの内部状態 h_3 が、そのままDecoder部に渡されることが示されている。入力シーケンスの情報のエッセンスが、この内部状態 h_3 に凝縮されていると考えればいい。



- AutoencoderのDecoder部が、圧縮された情報から元の情報を復元しようとするように、ここでは、その情報から、「同じ意味」を持つ、別の言語の文章を復元しようとする。

文の意味の分散表現
8,000次元の状態から
なる384M個のパラメーター



Attention Mechanism

新しいGoogle翻訳のシステムは、全く新しいアイデアに基づいてスクラッチから作り上げられたものではない。その基本的なアイデアは、2016年の5月にarXivに投稿された次の論文に多くを負っている。それは、先の、Ilya Sutskever らのシステムを発展させたものだ。

Bahdanau, D., Cho, K., and Bengio, Y. “**Neural machine translation by jointly learning to align and translate**” <https://goo.gl/HZxbNH>

Bahdanau, D., Cho, K., and Bengio, Y. “**Neural machine translation by jointly learning to align and translate**” <https://goo.gl/HZxbNH>

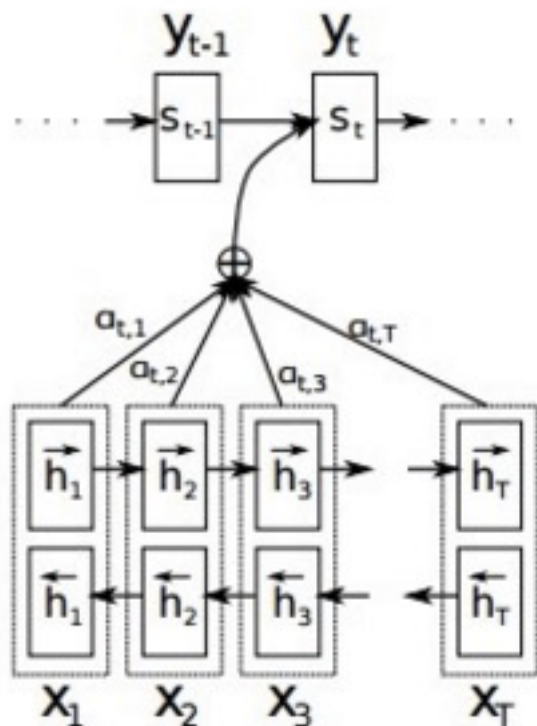
「近年、ニューラル機械翻訳として提案されたモデルは、多くの場合、Encoder-Decoderのファミリーに属している。そこでは、ソースの文が固定長ベクトルにエンコードされ、そこからデコーダが 翻訳文を生成する。この論文では、**固定長ベクトルの使用が、この基本的なEncoder/Decoderアーキテクチャの性能を改善する上でのボトルネックになっている**と推論し、モデルに自動的に、ターゲット・ワードを予測するのに重要なソース・文の一部分について、(ソフト)検索を可能とすることによって、これを拡張することを提案する。その際、これらの部分を明示的にハードセグメントとして形成する必要はない。」

固定長ベクトルがボトルネック

- 先に見た、Ilya Sutskever らの翻訳システムでは、翻訳されるべき文は、Encoderで、一旦、ある決まった大きさの次元（例えば8000次元）を持つベクトルに変換される。このベクトルからDecoderが翻訳文を生成する。入力された文が、長いものであっても短いものであっても、途中で生成され以降の翻訳プロセスすべての出発点となるこのベクトルの大きさは同じまま。このシステムでは、長くても短くても入力された文全体が、一つの固定長のベクトルに変換されるのだ。
 - 確かに、そこは翻訳の精度を上げる上でのボトルネックになりうる。事実、Ilya Sutskever らのシステムでは、文の長さが長くなるにつれて、翻訳の精度が低下されるのが観察されるという。
-

この論文の基本的アイデア

- 文全体に一つの固定長のベクトルを割り当てるのではなく、Decoderが翻訳時に、ソース・文の一部を改めて見直して、その部分から提供される情報を翻訳に生かそうということだ。



$a_{3,2}$ が大きい場合、これは、Decoderがターゲット文の第3の単語を生成しながら、ソース文の第2の状態に多くの注意を払うことを意味する。

人間と機械翻訳のギャップに橋をかける Googleのニューラル機械翻訳システム

***"Google's Neural Machine Translation
System: Bridging the Gap between Human
and Machine Translation"***

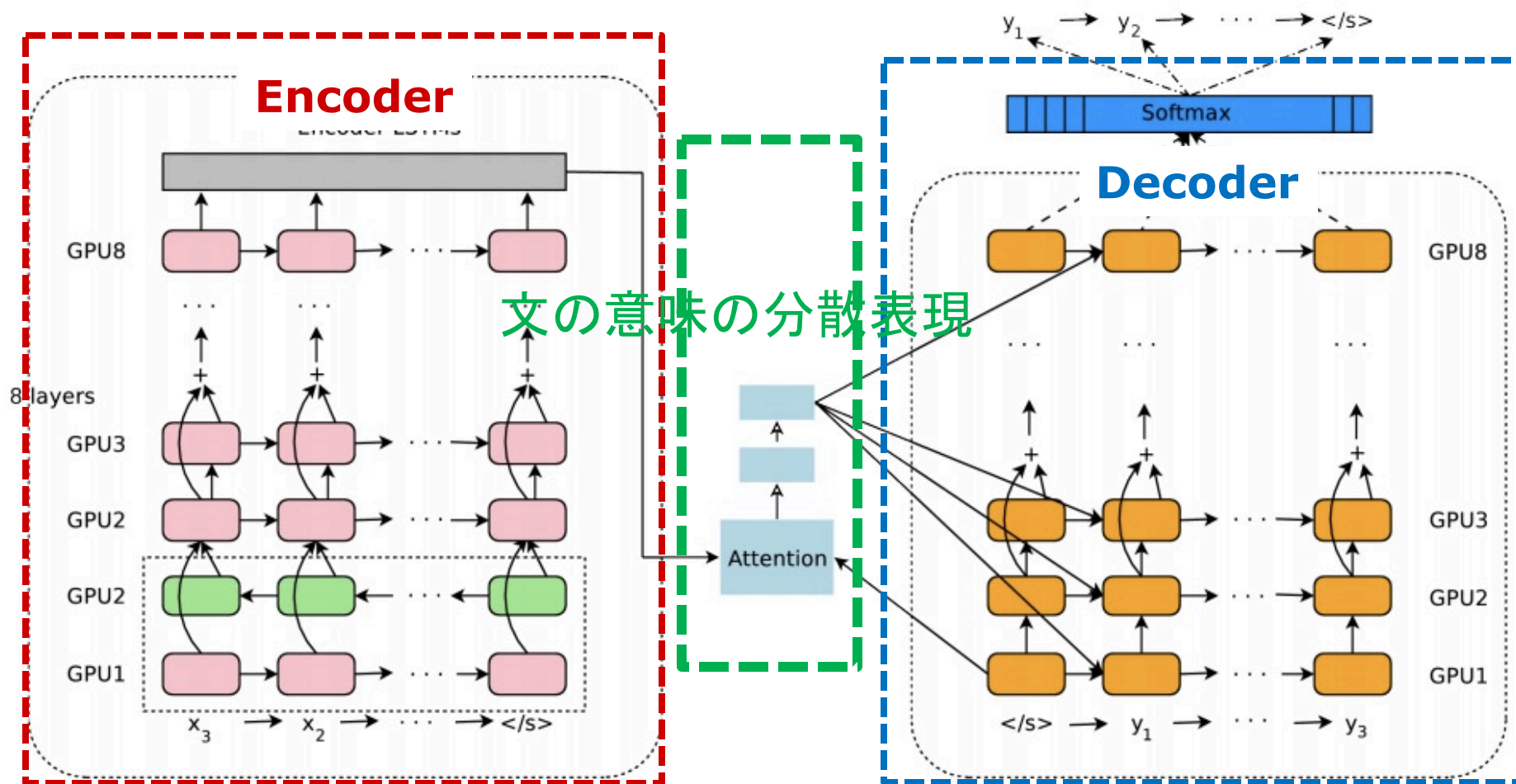
Yonghui Wu et al.

<https://goo.gl/YqIAAW>

2016年

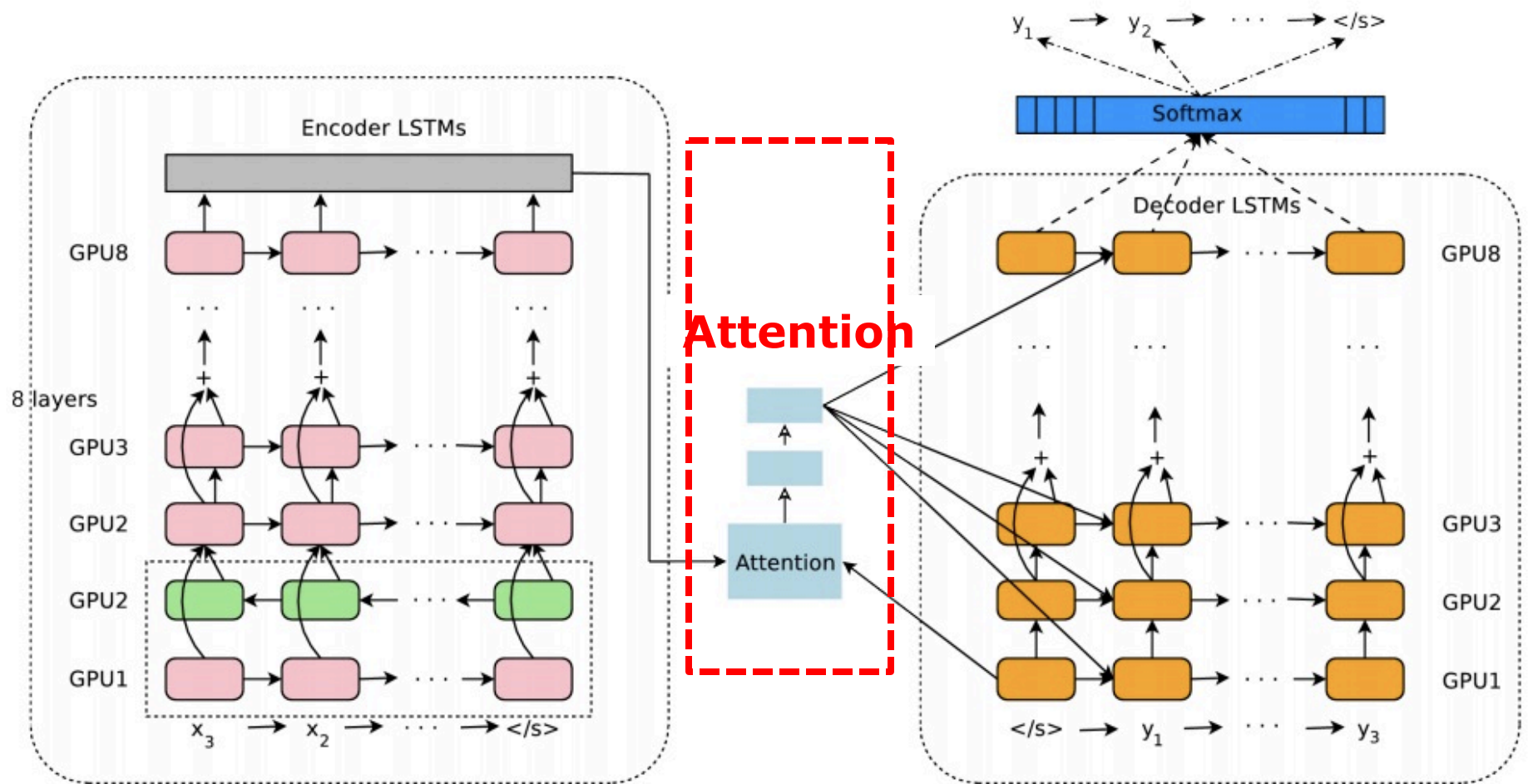


Encoder / Decoder



左側に、LSTMを8段重ねにした Encoder LSTMがあり、
右側には、同じくLSTMを8段重ねにした Decoder LSTMがある。

Attention Mechanism



EncoderとDecoderの間に、Attentionと記された領域がある。
ここからの出力Attention Context は、Decoderのすべての
ノードに供給されている。

訓練に利用されたコーパス

- WMT14の英語(En) <-> フランス語(Fr)データセットには、3,600万の文のペアが含まれている。
 - WMT14の英語(En) <-> ドイツ語(De)データセットには、500万の文のペアが含まれている。
 - Googleは、内部に、英語 <-> 日本語(Ja)、英語 <-> 韓国語(Ko)、英語 <-> スペイン語(Es)、英語 <-> ポルトガル語(Pt) 等々の多くのデータセットを持っているが、**その規模は、先のWMTのデータセットより、2～3桁大きいという。**
 - Googleニューラル機械翻訳では、GPU100個を使って、フルトレーニングには最大1,000万ステップ、収束までには3週間かかることがあるという。
-

三つのモデルのまとめ

基本的な観察

文と意味の「構成性(compositionality)」

- 文は文法に従って語から構成される
- 文の意味は、語の意味に依存する
 - 文法的に正しいが意味がない文もある
Colorless green ideas sleep furiously.
- 文の意味は、文の文法的構成に依存する
 - 二つの意味を持つ、あいまいな文
I saw a man with a telescope.

意味の「同一性」 / 意味の共通表現の存在

- 言語が異なっても、「同じ」意味を持つ語が存在する。
- 言語が異なっても、「同じ」意味を持つ文が存在する。

現在の主要な自然言語処理の実装は 文法・文の構成性を考慮していない

	語の意味の 「表現」	文の構成性 Grammar / Syntax	文の意味の 「表現」 Semantics
Intentモデル	X	X	X
Entityモデル	Entityの Ontology	X	Action Typeの Property
機械翻訳モデル	X	X	多次元ベクトル による分散表現 (暗黙の利用)

基本的な観察

文と意味の「構成性(compositionality)」

- 文は文法に従って語から構成される
- 文の意味は、語の意味に依存する
 - 文法的に正しいが意味がない文もある
Colorless green ideas sleep furiously.
- 文の意味は、文の文法的構成に依存する
 - 二つの意味を持つ、あいまいな文
I saw a man with a telescope.

意味の「同一性」 / 意味の共通表現の存在

- 言語が異なっても、「同じ」意味を持つ語が存在する。
- 言語が異なっても、「同じ」意味を持つ文が存在する。

文法と意味の表現を意識したモデルも存在する

	語の意味の 「表現」	文の構成性 Grammar / Syntax	文の意味の 「表現」 Semantics
Word2Vec	多次元ベクトル による分散表現	—	—
IBM Watson内 のプロジェクト	辞書項目	English Slot Grammar	論理式
Combinatory Categorial Grammar	辞書項目	Categorial Grammar	型付きラムダ式
DisCoCat	多次元ベクトル による分散表現	Pregroup Grammar	多次元ベクトル による分散表現

第三部で紹介

文法と意味の表現を意識したモデルも存在する

	語の意味の 「表現」	文の構成性 Grammar / Syntax	文の意味の 「表現」 Semantics
Word2Vec	多次元ベクトル による分散表現	—	—
IBM Watson内 のプロジェクト	辞書項目	English Slot Grammar	論理式
Combinatory Categorical Grammar	辞書項目	Categorical Grammar Curry—Howard対応	型付きラムダ式
DisCoCat	多次元ベクトル による分散表現	Pregroup Grammar Functor Semantic	多次元ベクトル による分散表現

第三部で紹介

意味の表現をめぐる様々な試み I

-- Deep Learningの側から --

Word2Vec

Scene Graph

Image Sentence Mapping

Word2Vec 語の「意味ベクトル」

"Linguistic Regularities in Continuous Space Word Representations"

Tomas Mikolov et al.

<http://goo.gl/j25y8s>

2013年

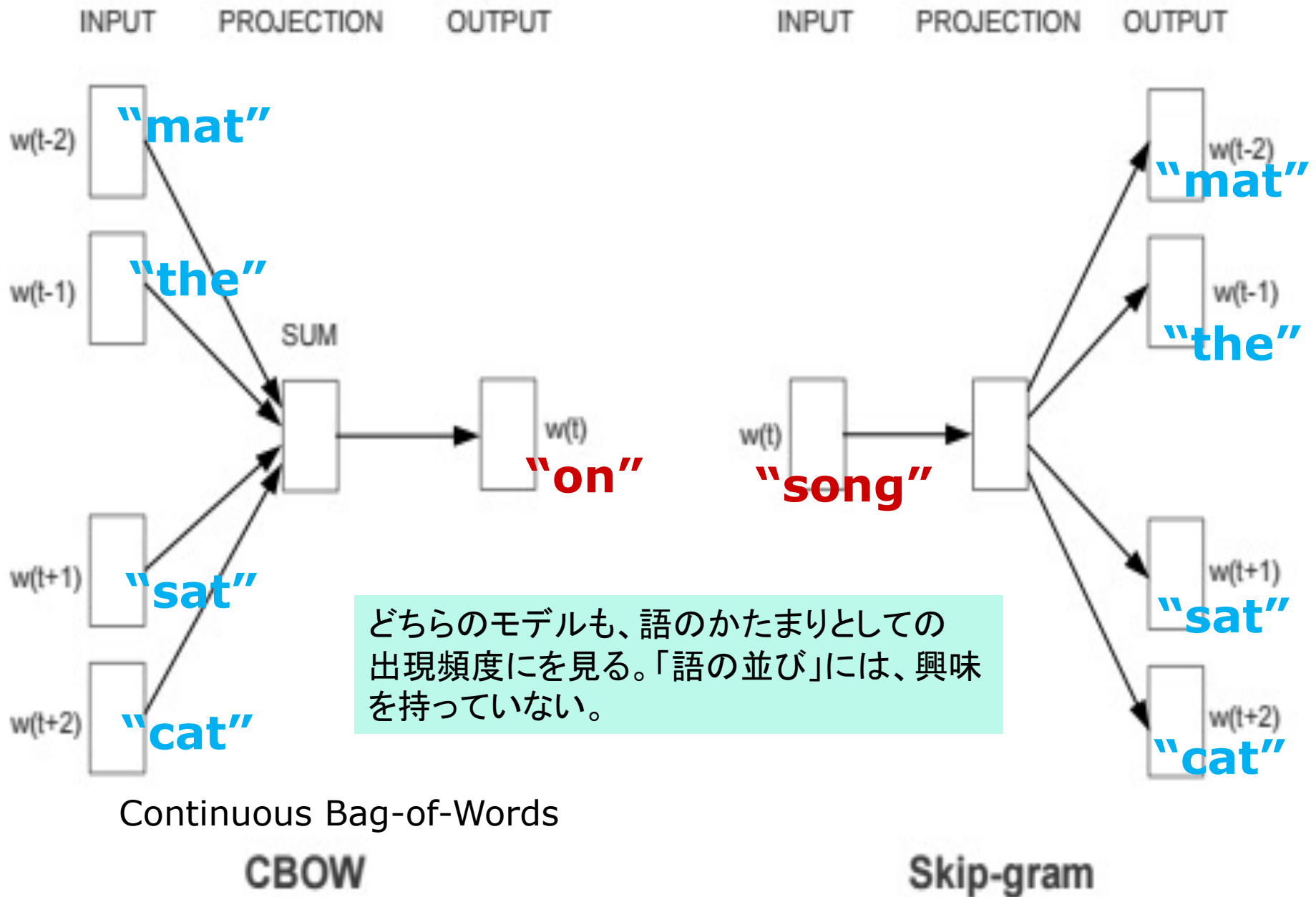


Word2Vec

- 2013年に、GoogleのTomas Mikolovらは、語が埋め込まれたベクター空間が、言語学的に(文法的にも、意味論的にも)面白い性質を持っていることを発見する。
 - “We find that these representations are surprisingly good at capturing syntactic and semantic regularities in language, and that each relationship is characterized by a relation-specific vector offset.” (Linguistic Regularities in Continuous Space WordGoog Representations <http://research.microsoft.com/pubs/189726/rvecs.pdf>)
 - Jeff Deanも、この論文に注目して、共同の作業が始まる。
-

Word2Vec

- Google Codeに、オープンソースとして公開され、
<https://code.google.com/p/word2vec/> 大きな関心を集める。
 - ここで使われているモデルは、次の二つ。
 - **CBOW(Continuous Bag-of-Word) モデル**
複数の語の集まりから、一緒に出現しそうな一つの語の確率を調べる。
 - **Skip-gram モデル**
一つの語が与えられた時、一緒に出現しそうな複数の語の確率を調べる。
-



どんな語が、与えられた語の近くに埋め込まれるか？

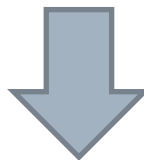
FRANCE	JESUS	XBOX	REDDISH	SCRATCHED	MEGABITS
AUSTRIA	GOD	AMIGA	GREENISH	NAILED	OCTETS
BELGIUM	SATI	PLAYSTATION	BLUISH	SMASHED	MB/S
GERMANY	CHRIST	MSX	PINKISH	PUNCHED	BIT/S
ITALY	SATAN	IPOD	PURPLISH	POPPED	BAUD
GREECE	KALI	SEGA	BROWNISH	CRIMPED	CARATS
SWEDEN	INDRA	PSNUMBER	GREYISH	SCRAPED	KBIT/S
NORWAY	VISHNU	HD	GRAYISH	SCREWED	MEGAHERTZ
EUROPE	ANANDA	DREAMCAST	WHITISH	SECTIONED	MEGAPIXELS
HUNGARY	PARVATI	GEFORCE	SILVERY	SLASHED	GBIT/S
SWITZERLAND	GRACE	CAPCOM	YELLOWISH	RIPPED	AMPERES

どんな語が、与えられた語の近くに埋め込まれるか？

- 似た意味を持つ言葉は、似たベクトルを持つ。
- 似た言葉で置き換えても、正しい文は、正しい文に変わる。

“a **few** people sing well”

正しい文



“a **couple** people sing well”

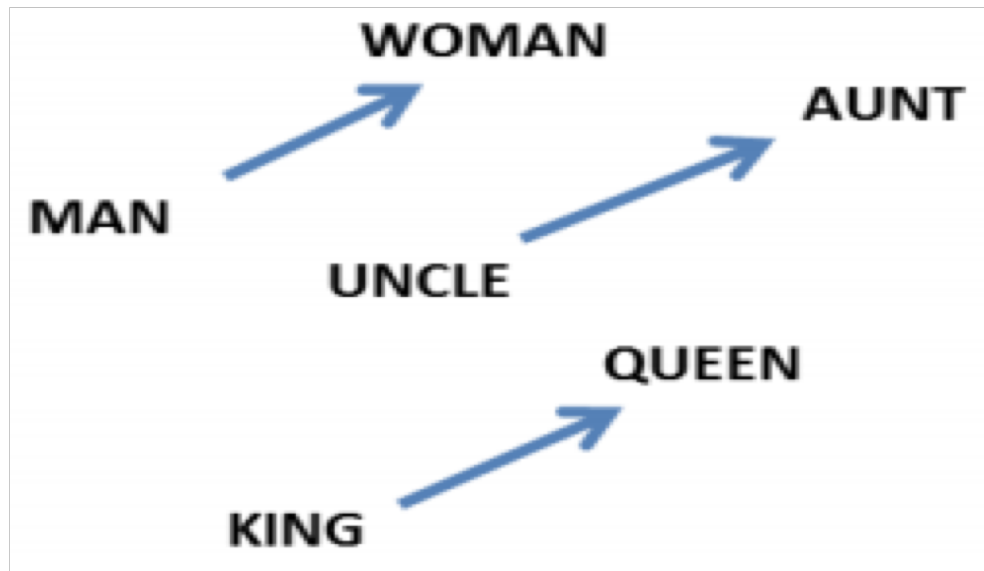
正しい文

- 意味が似ていなくても、同じクラスの言葉で置き換えても、正しい文は、正しい文に変わる。

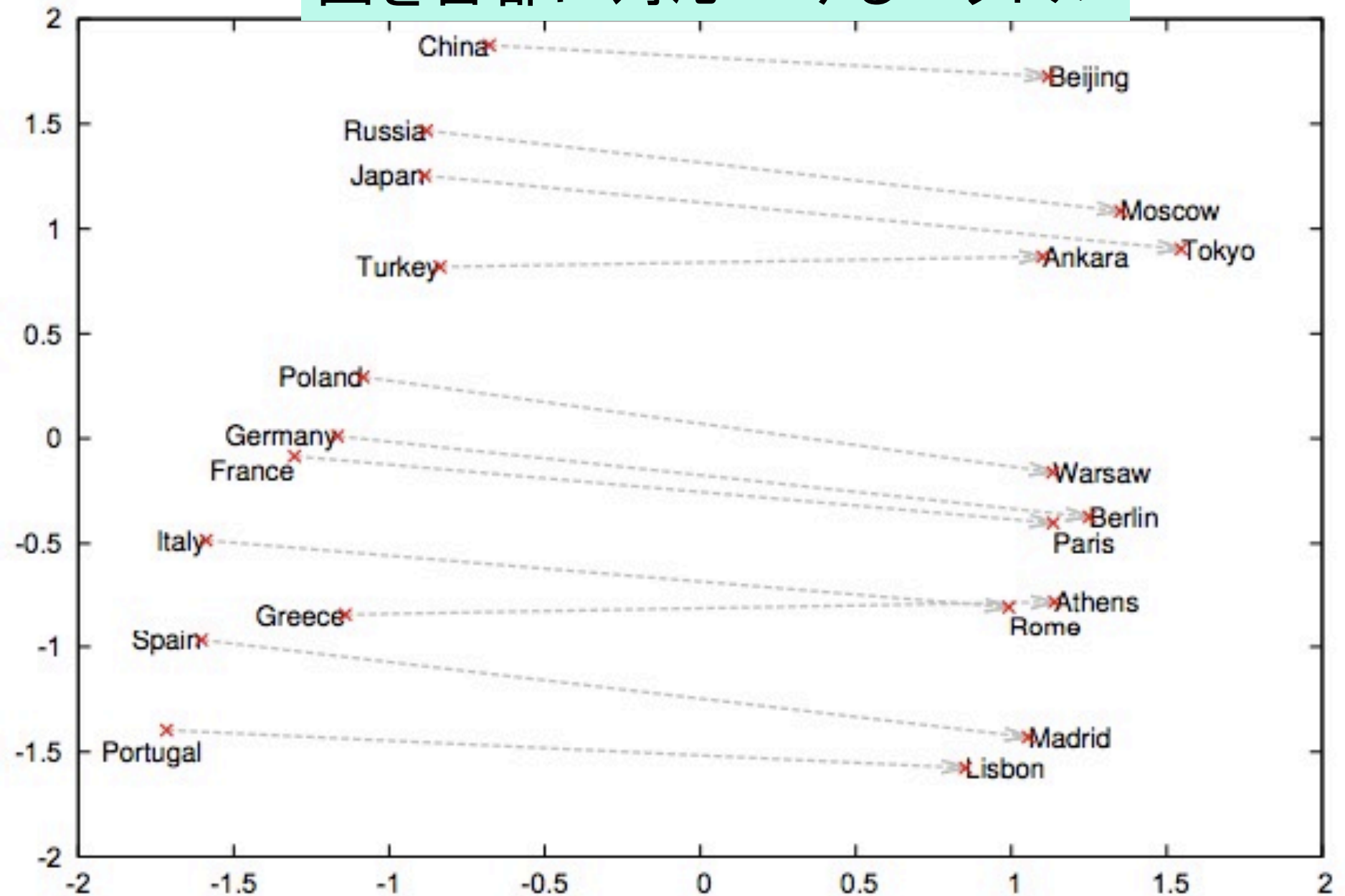
“the **wall** is **blue**” → “the **ceiling** is **red**”

意味を変換するベクトルは共通？

- Word Embeddingは、もっと面白い性質を持つ。下の図のように、男性から女性へのベクトルがあるように見える。
- $W(\text{"woman"}) - W(\text{"man"}) \simeq W(\text{"aunt"}) - W(\text{"uncle"})$
 $W(\text{"woman"}) - W(\text{"man"}) \simeq W(\text{"queen"}) - W(\text{"king"})$



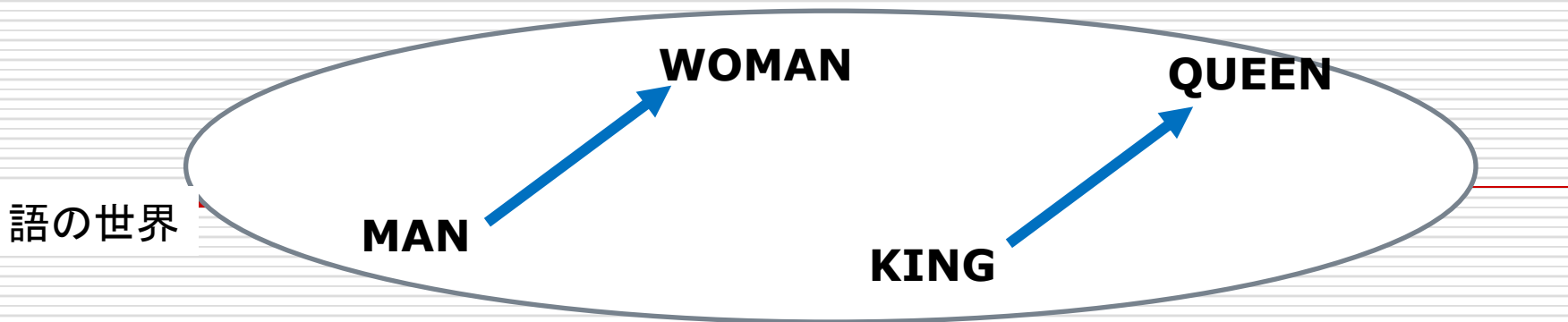
国を首都に対応づけるベクトル



Word2Vecと語の意味の表現

Word2Vecで留意すべきこと

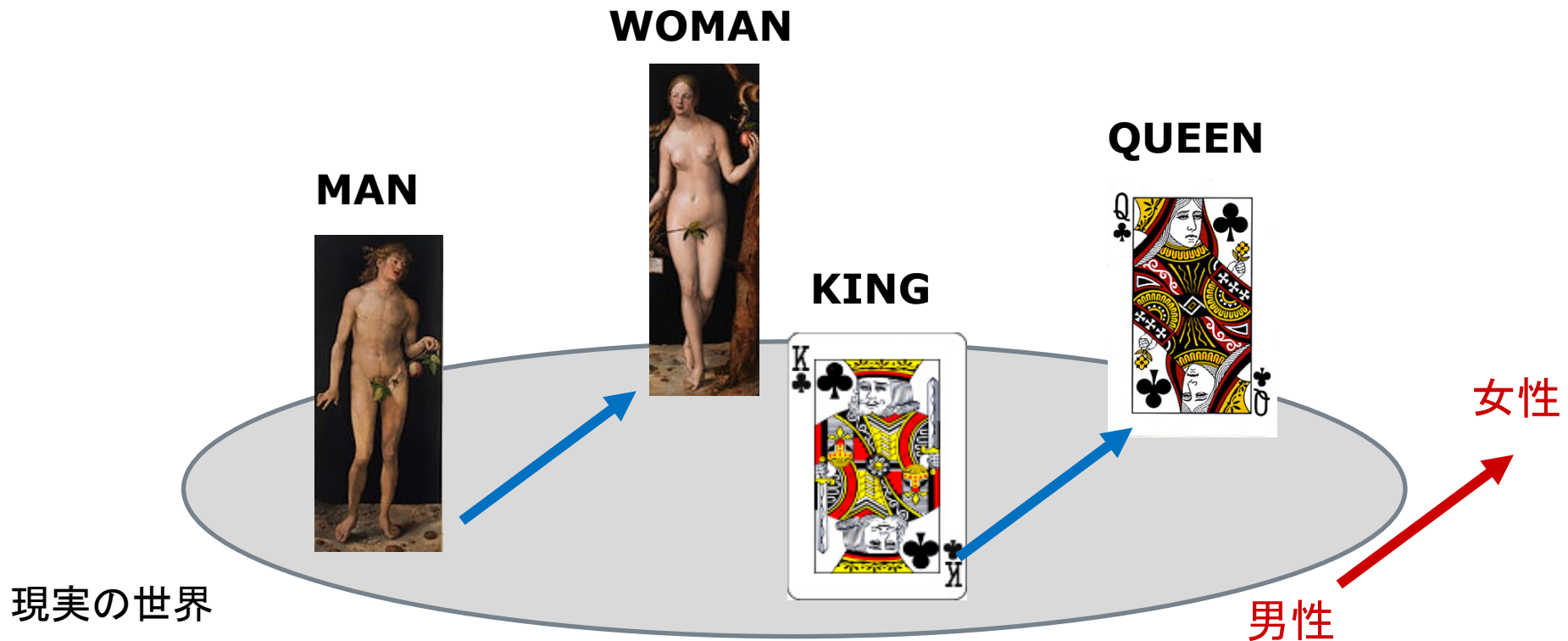
- Word2Vecが捉えようとしているのは、「語の意味」の対応物である。それは、直接的には、「文の意味」をターゲットにしたものではない。(言語学的には、Wordは、後天的に学習される Lexical Itemである。)
- Word2Vecが捉える語の諸特徴は、学習に利用されるデータセットに依存している。学習用のデータセットの拡大、あるいは変更依存しないで、これらの諸特徴が収束することを、このアプローチ自身は保証できない。
- もし、Word2Vecが、学習用データセットに依存しない語の諸特徴を捉えているのなら、その根拠は、データセットの外に求められなければならない。

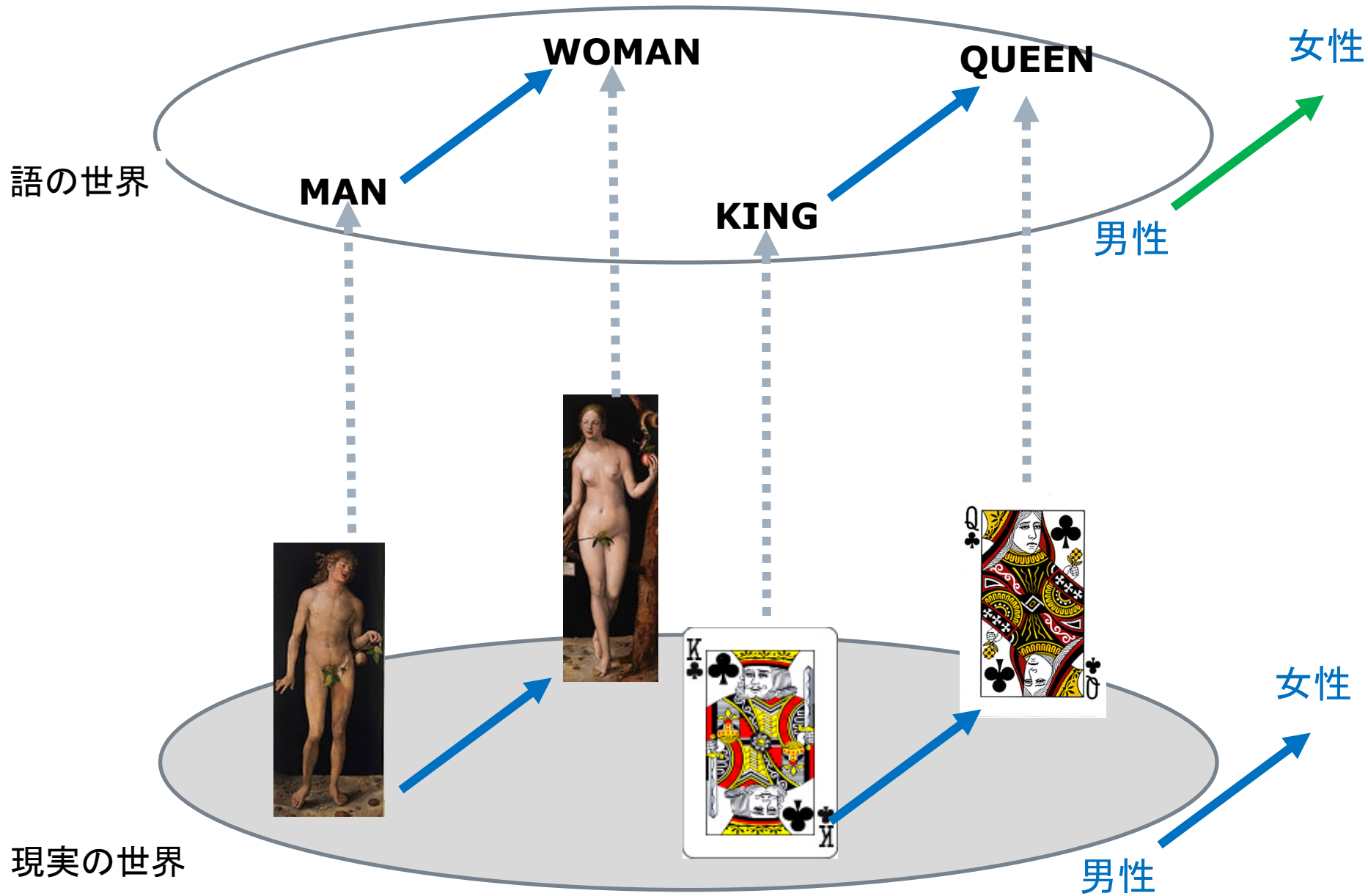


Word2Vecでは、MANからWOMENへのベクトルが、KINGからQUEENのベクトルと一致するという。(上図)

この事実のもっとも素朴な説明は、**現実の世界**で、MAN, WOMENとKING, QUEENの関係を考える事である。

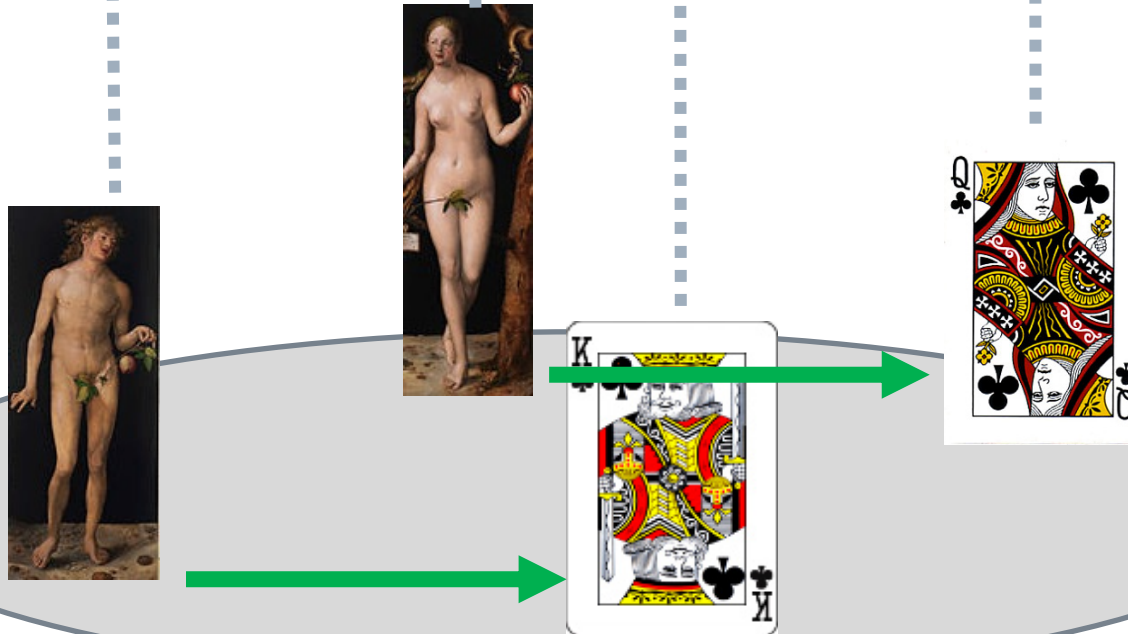
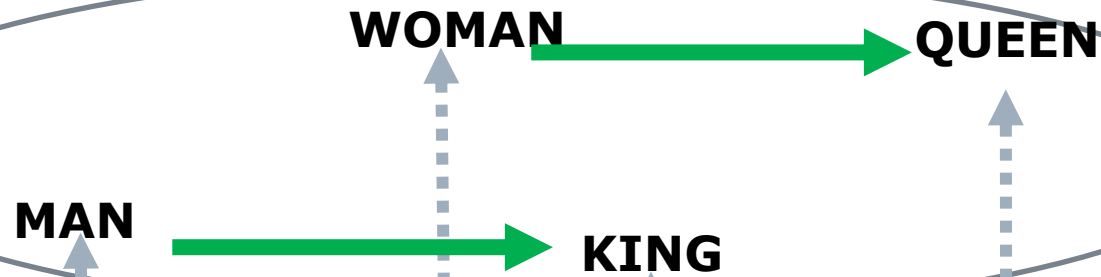
現実の世界で、MAN対WOMENとの関係が、KING対QUEENの関係に一致する「なんらかの特徴」が、存在するか考えてみよう。この例の場合には、「男性」対「女性」という関係が、すぐに見つかる。





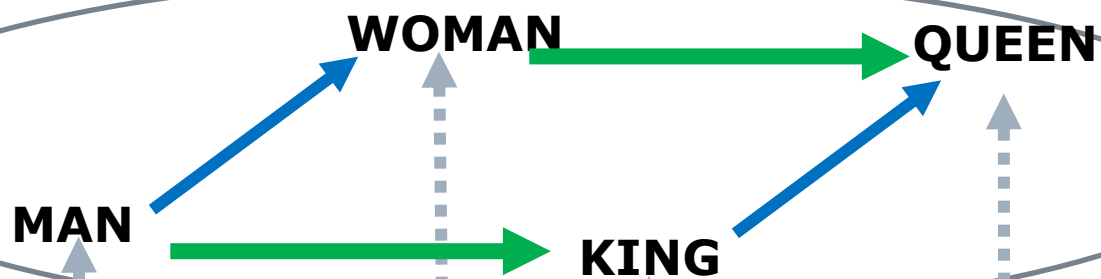
こうした「意味ベクトル」も存在するはず

語の世界

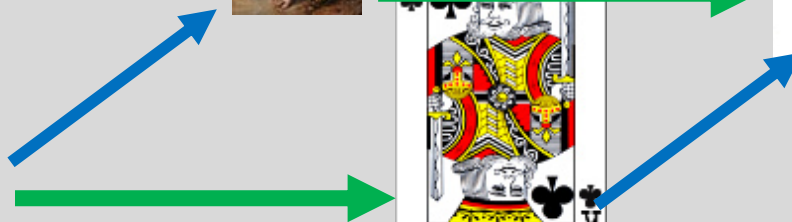


現実の世界

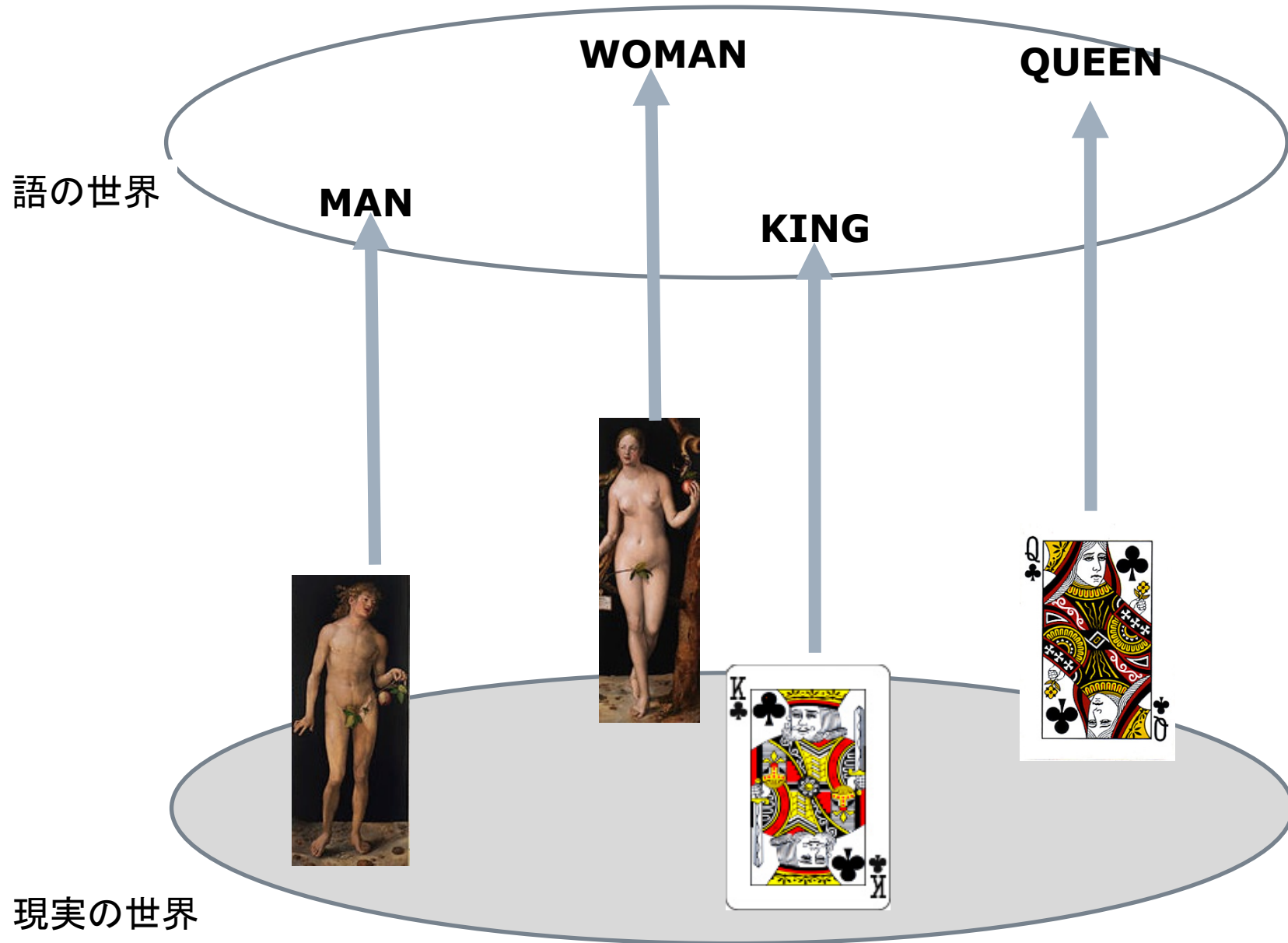
語の世界



現実の世界



「語の意味」の素朴なモデル



Word2Vecの表すもの

- Word2Vec が表しているものを、解釈するには、二つの語のなすベクトルを考えるのがわかりやすい。それは二つの語の関係であり、その限りでの関係の意味だ。それは、一つの語の意味を表すものではない。
- ただ、先に見たように、「現実の世界」と「語の世界」の対応を考えると、別の見方もできる事に気づく。すなわち、「現実の世界」の事物に、「語の世界」の語を対応付けること、事物に名前をつけることが、語に意味を与えていると。逆に、語は、基本的には、現実の世界の何かを参照することで、意味を獲得するのだと。
- こうした見方で、多くの言語の辞書的語彙項目には、比較的簡単な対応関係があることを示すことができる。それは、「同じ意味を持つこと」のプリミティブな表現を与える。

Scene Graph

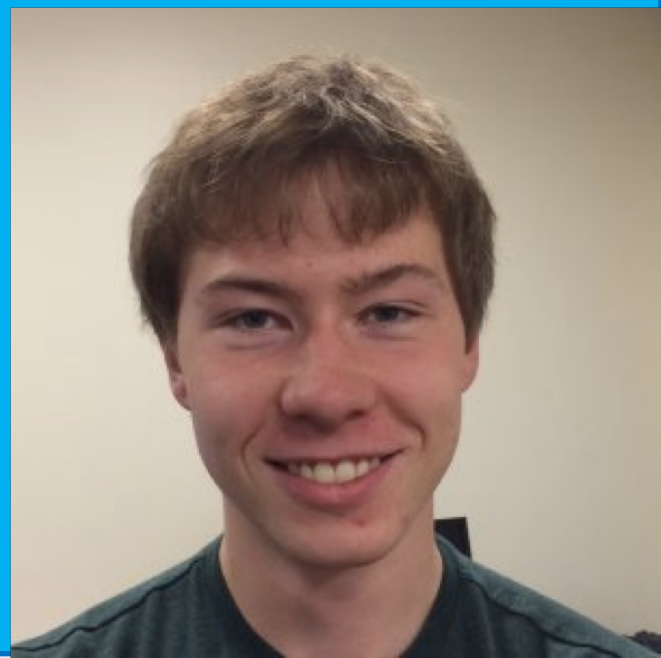
Image Retrieval using Scene Graphs

Justin Johnson et al.

<http://goo.gl/xAmz6E>

<http://goo.gl/3GCrXM> (samples)

2015年



目標: 自然言語を用いた画像検索

man holding fish and wearing hat on white boat



(a) Results for the query on a popular image search engine.



(b) Expected results for the query.

なぜ、scene graphsか？

シーンの内容を表現する、もう一つ別の選択は、明らかである。それは、次の例のように、**自然言語**を用いることである。

"A blonde white girl is standing in front of an orange cone on a lined tennis court and is holding a long heavy yellow wide racket that has a black handle. The girl is wearing a white shirt; there is a bent arm in front of the shirt and another bent arm beside the first. There is a round yellow logo on the shirt, and the logo is beside hands that are on the handle of the racket. There is a black fence behind the girl, and the girl has brown eyes above a closed mouth. There are butterflies barrettes in long blonde hair, and the hair is in a ponytail."

なぜ、scene graphsか？

画像の意味表現にEntityモデルを採用

- 画像検索に、こうした自然言語での記述を利用するためには、次のようなことが必要になる。
 - テキスト中の相互参照関係の解決。
 - 構造化されていないテキストを、構造化されたtupleに変換することで、関係の抽出を実行する。
 - このtupleを構成する実体を、テキストで記述された領域の中に基礎づける。
- こうした処理のパイプラインを実行することは、条件を限ってもチャレンジングな課題である。そして、先に見たような詳細なテキストでは、スケールしない。
- 注意深く設計されたユーザーインターフェースの下では、専門家ではない素人でも、どんな複雑なシーングラフでも、簡単に作れることを、我々は発見した。

Scene Graph

- 自然文の生成ではなく、画像を解析して、それが表すシーンを要素・関係・属性等でグラフ化する。
we use Amazon's Mechanical Turk (AMT) to produce a human-generated scene graph.
 - 行っているのは、
 - (1) オブジェクトの検出と分類。
 - (2) 検出した二つのオブジェクトの位置関係・状態の検知。
 - (3) オブジェクトの属性の検知である。
-



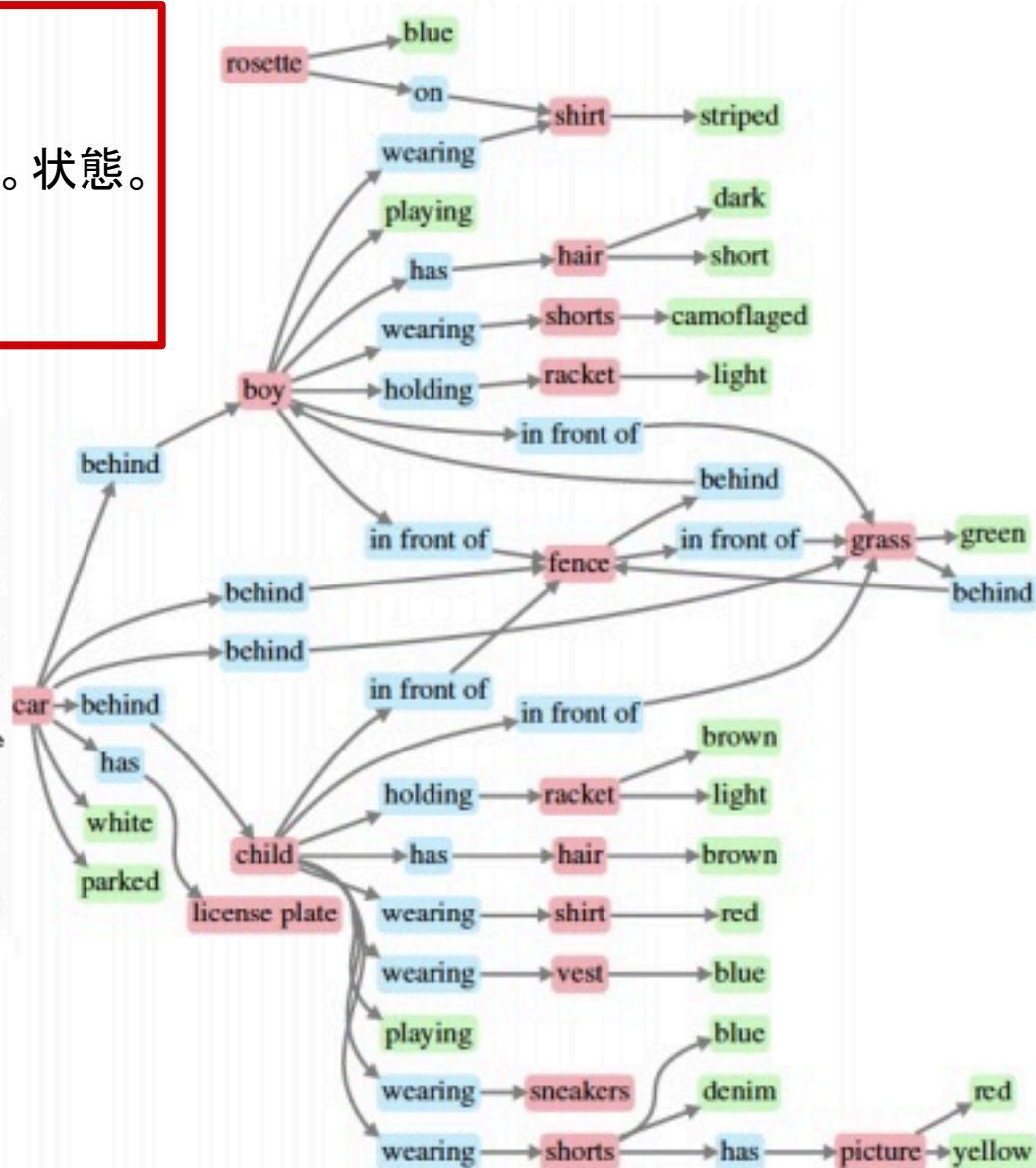
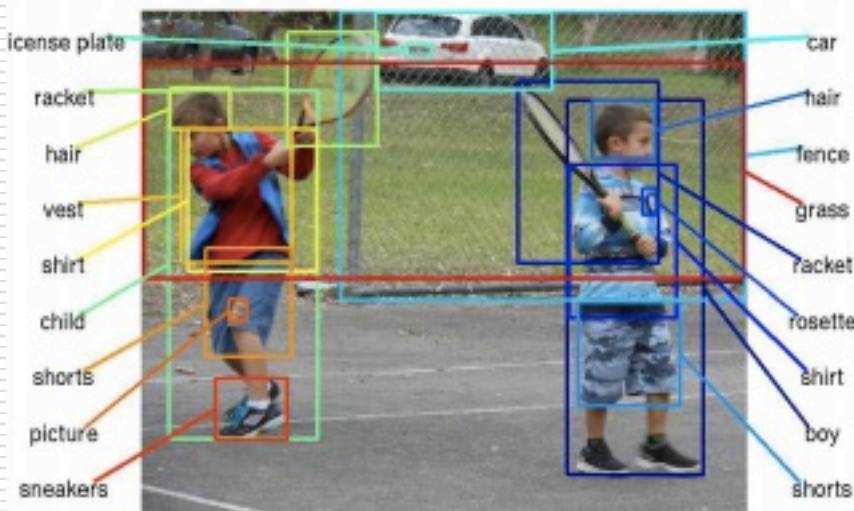
オブジェクト

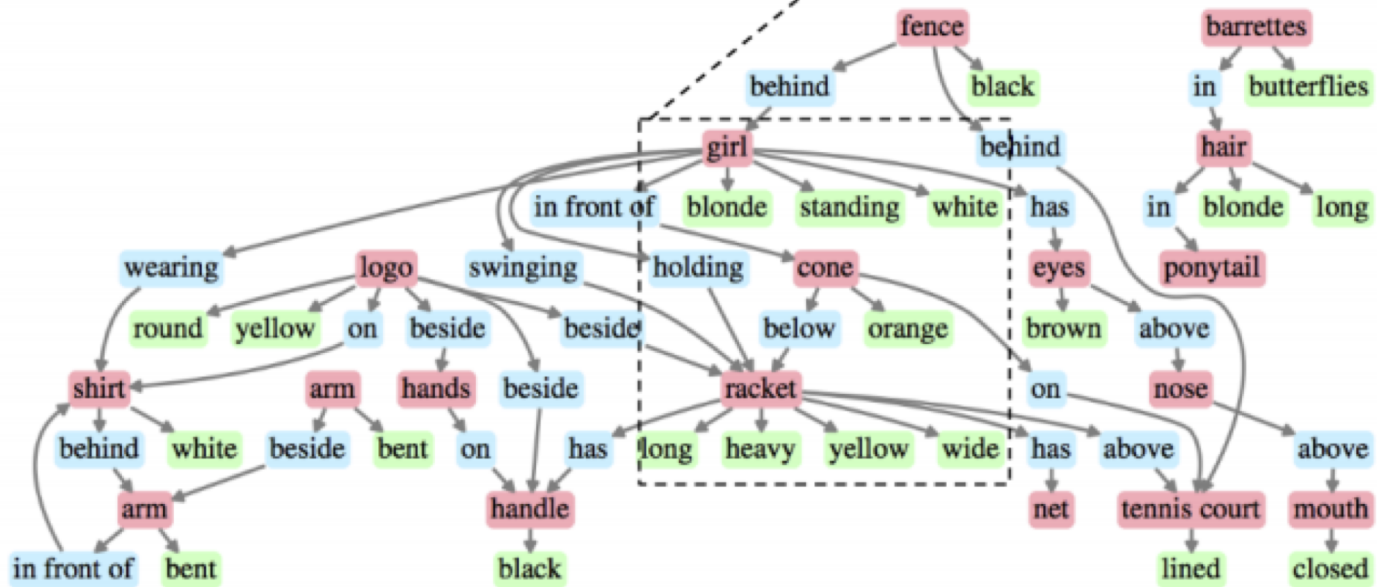
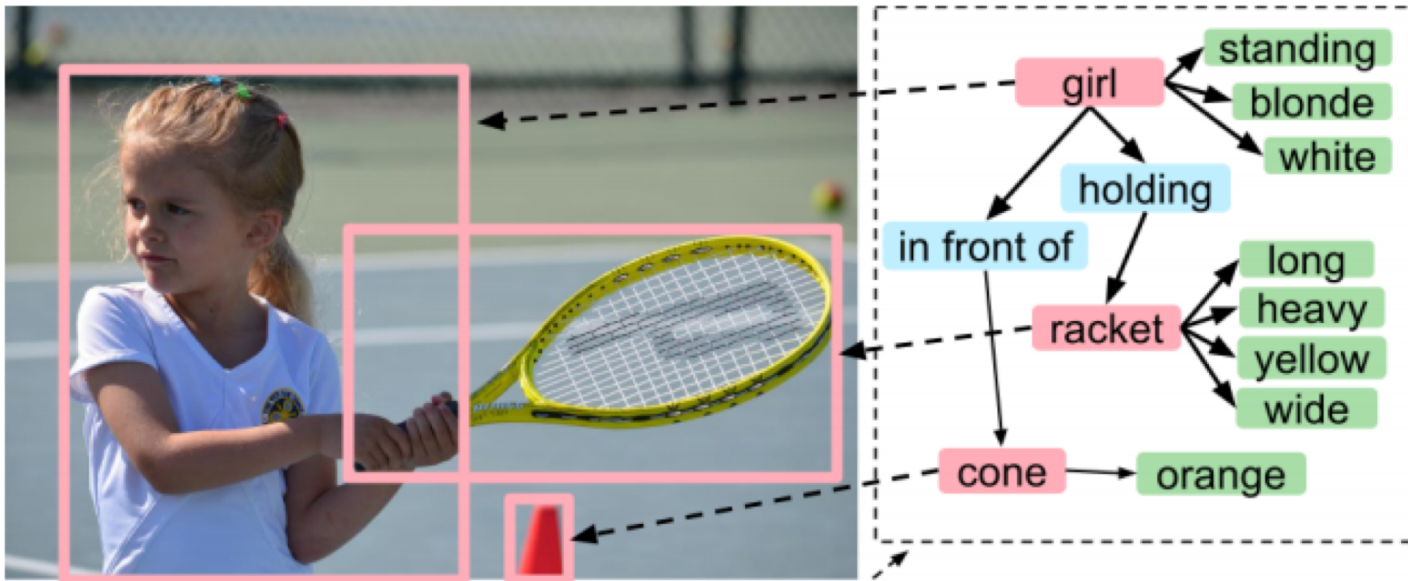


オブジェクト間の位置関係。状態。



オブジェクトの属性





Legend:



objects

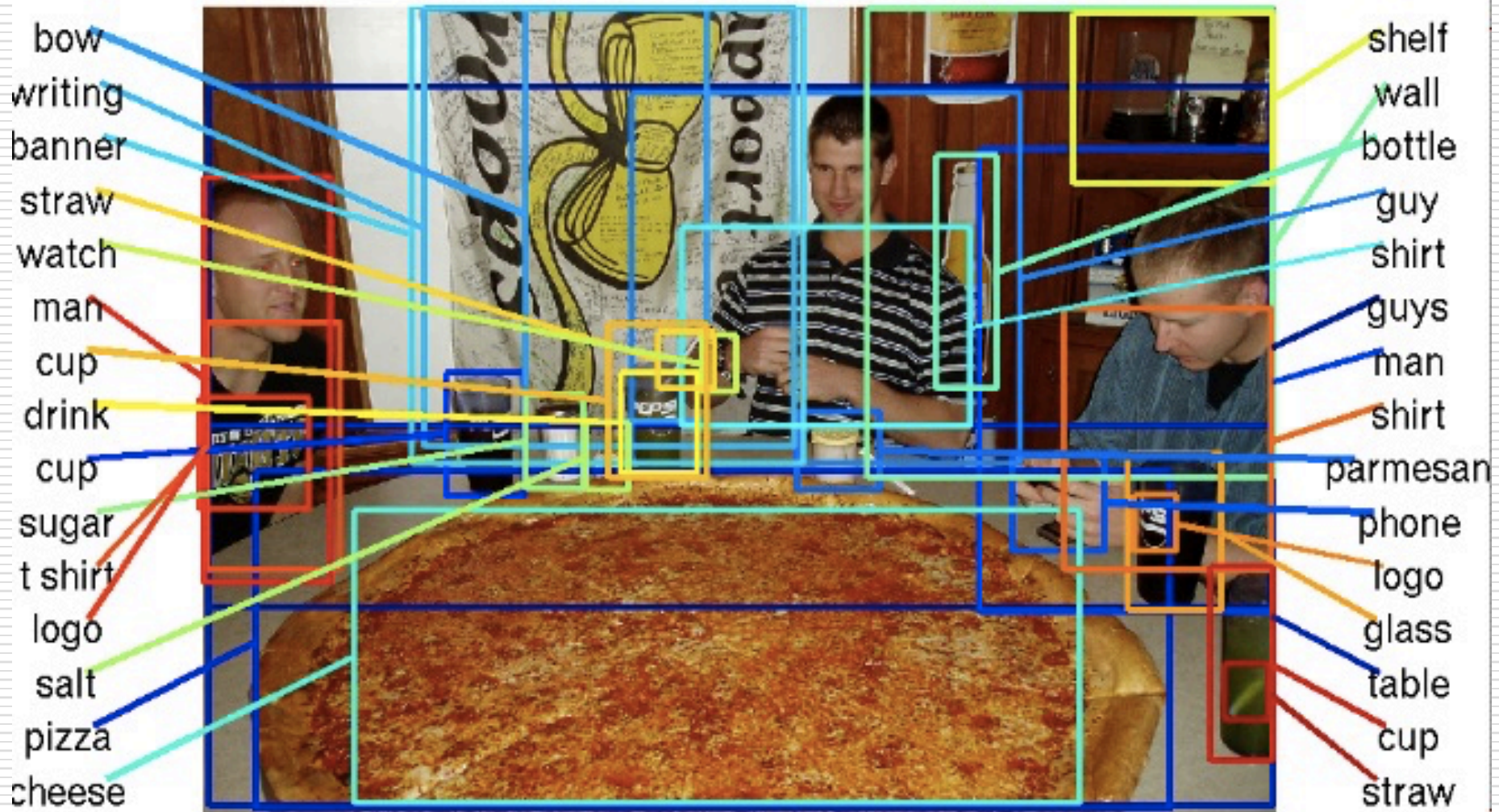


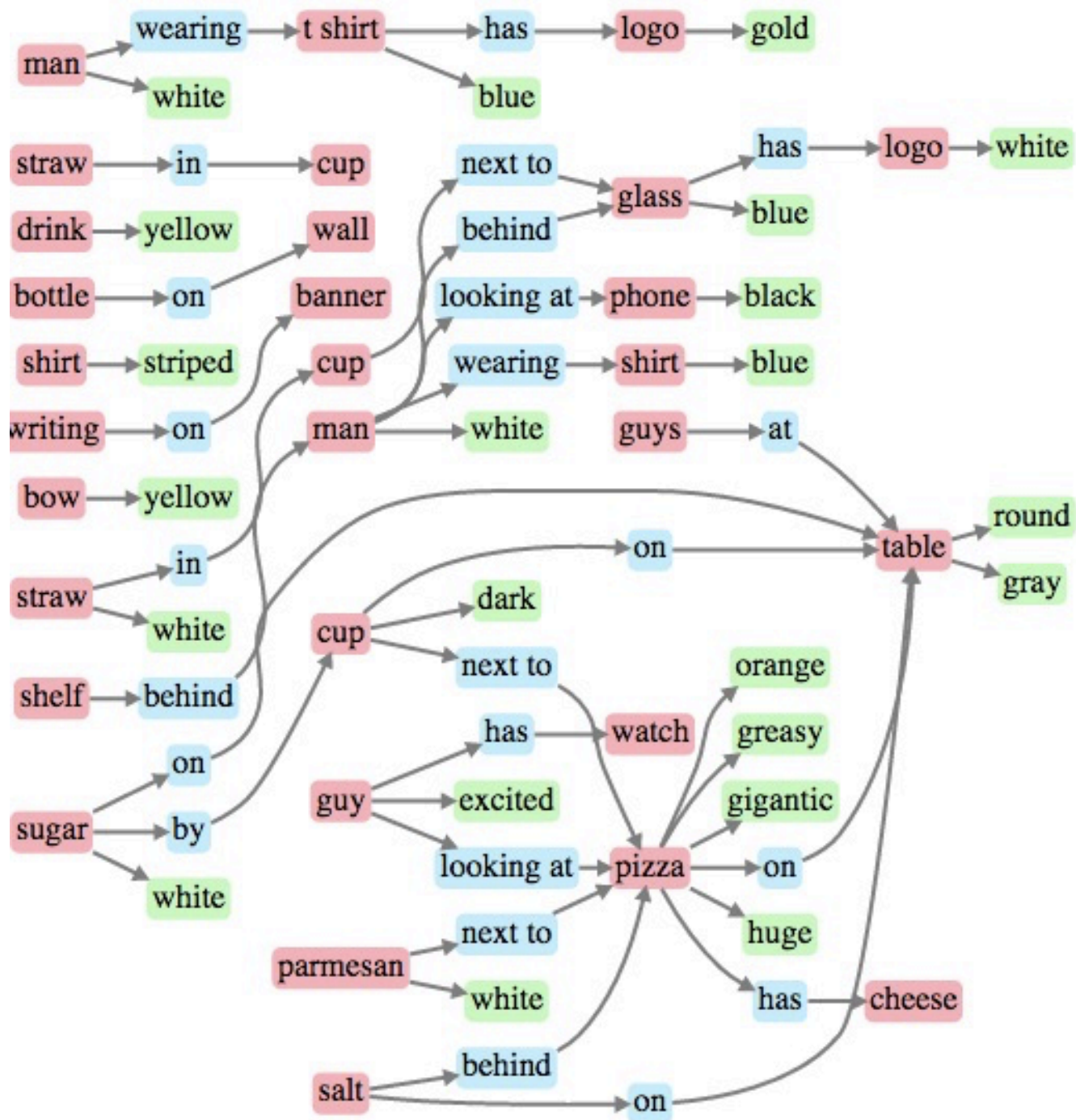
attributes



relationships

画像と、それから作られるscene graphの例





シーン・グラフから、自然言語によるシーンの記述を生成することは、容易である。



Black phone is on top of white, wooden desk. The desk is next to a clean white bed that has a black blanket and is next to a white table. The lamp is on a tan wall. The table is by the bed, which is next to the phone. The floor is under the bed, table, lamp and blanket.

黒い電話が白い木の机の上にある。机は、黒い毛布がある清潔なベッドの近くにあり、白いテーブルの近くにある。ランプは茶色の壁にある。テーブルはベッドに近く。電話に近い。床はベッド、テーブル、ランプ。毛布のの下にある。

Scene Graphと「概念のリソース」の存在

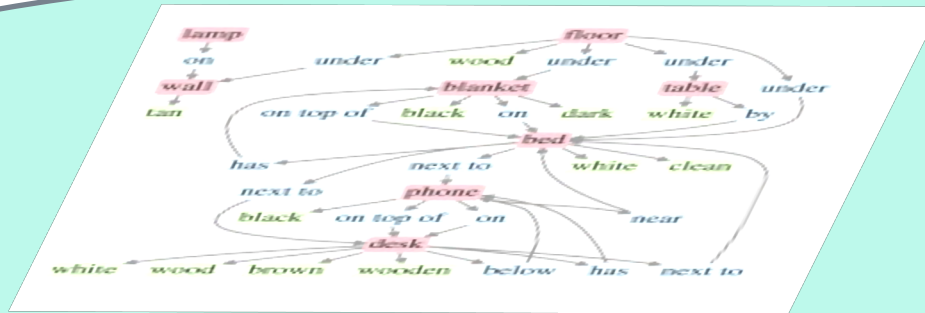
Scene graphの「意味」

- Scene graphで興味ふかいことは二つある。
- 一つは、人間なら、適切なツールのサポートがあれば、誰でも、しかもどんな複雑なscene graphでも作成できるということだ。これは、現在の機械にはできないことだ。
- もう一つは、生成されたscene graphを、比較的容易に「文」に変換できるということである。それは、現在の機械でもできそうである。(彼らは、それをしていないのだが)
- こうして変換された文は、画像の意味を表現している。

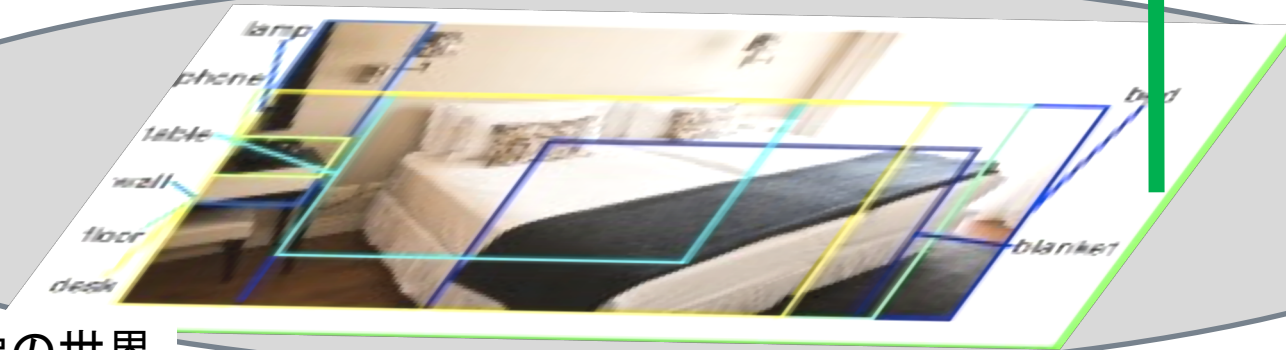
文の意味？

「黒い電話が白い木の机の上にある。机は、黒い毛布がある清潔なベッドの近くにあり、白いテーブルの近くにある。ランプは茶色の壁にある。テーブルはベッドに近く。電話に近い。床はベッド、テーブル、ランプ、毛布の下にある。」

文の世界



Scene Graph

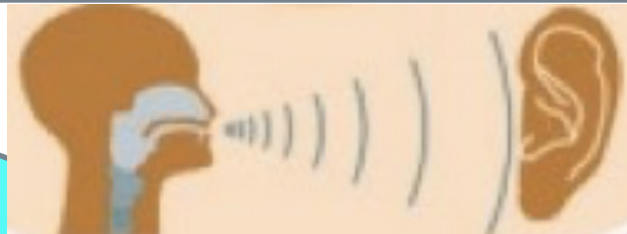


現実～画像の世界

「概念のリソース」の存在とその生得性

- Scene graph の作成者たちは、直接には文を生成するわけではない。その意味では、それは、人間の狭い意味での言語能力を直接に全面的に利用したものではない。
- この時、利用されているのは、Chomskyが言う「その上で、言語能力が描く概念のリソースが存在する。そして、言語の進化の研究が取り組むべき問題が存在する。すなわち、言語以前の、人間にも利用可能な概念のリソース」だと思ってい。
- それは、構造を持った、意味をになう文に近いものを生成する。

Minimalist Program の言語モデル



感覚-運動
インターフェース

外部化
Externalization

I-Language

帰納的計算

概念-思考
インターフェース

概念
Semantics



Scene Graph

オブジェクトと、
その関係・属性
の認識

ものの世界
の認識(多分、
動物も可能)

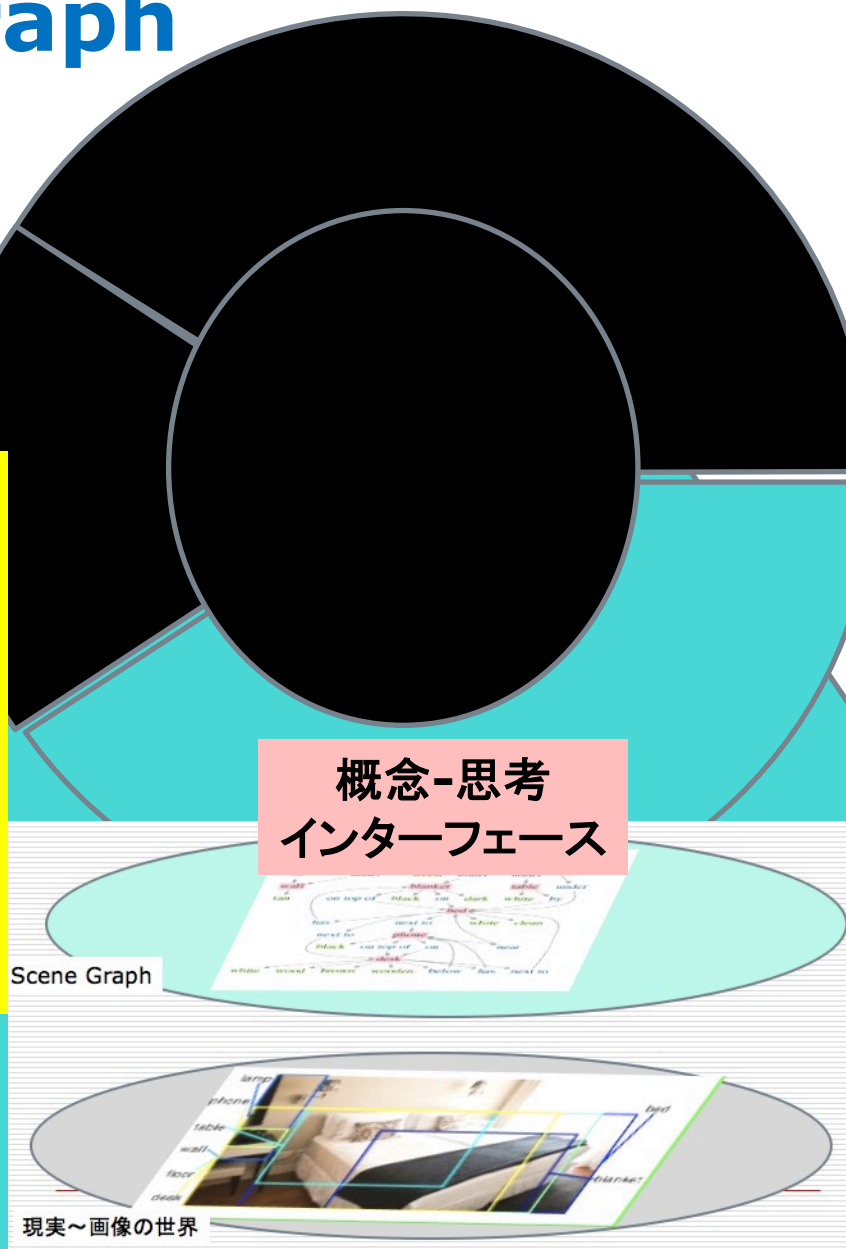
概念のリソース

Scene graphの
作成者たちは、
「概念・思考インター
フェイス」と、そ
の先の「概念のリ
ソース」を利用してい
る

概念-思考
インターフェイス

Scene Graph

現実～画像の世界



文章の一節から画像を検索する
画像の一部をフレーズに対応させる

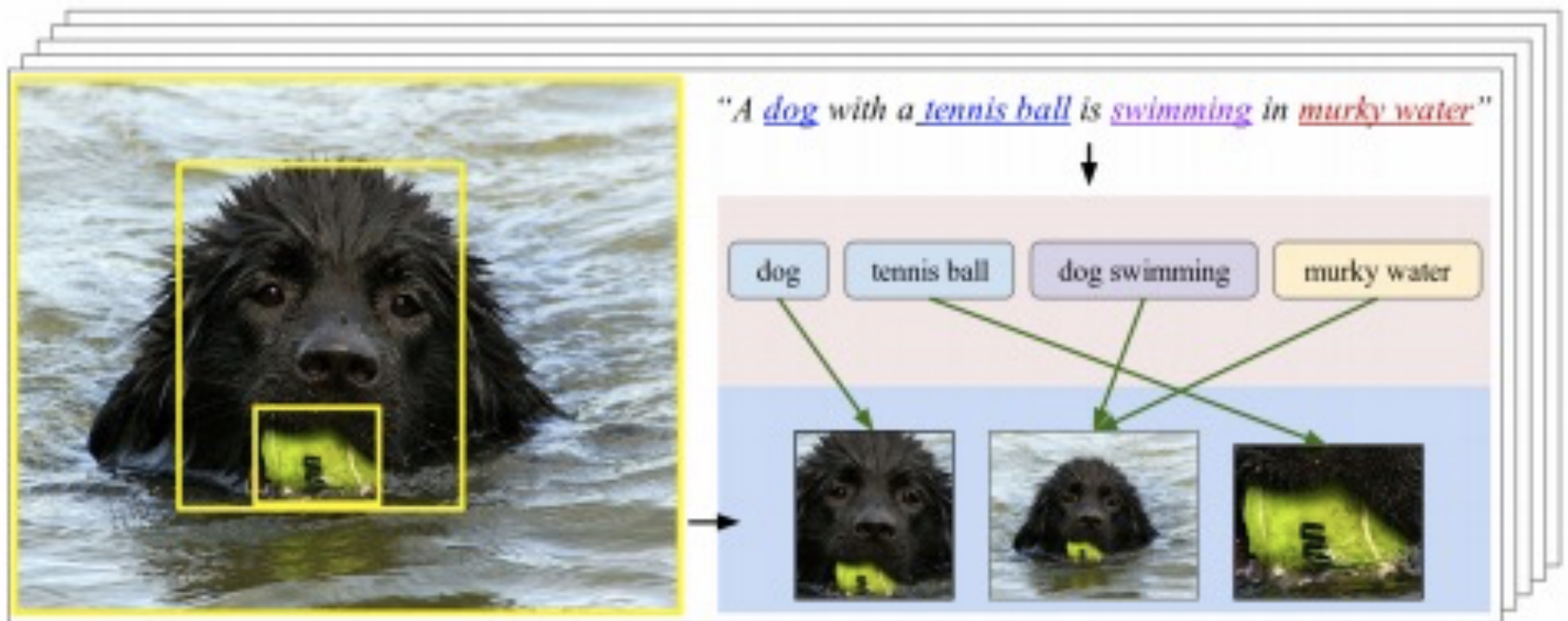
"Deep Fragment Embeddings for Bidirectional Image Sentence Mapping"

Andrej Karpathy et al.
<https://goo.gl/neBTk6>

2014年

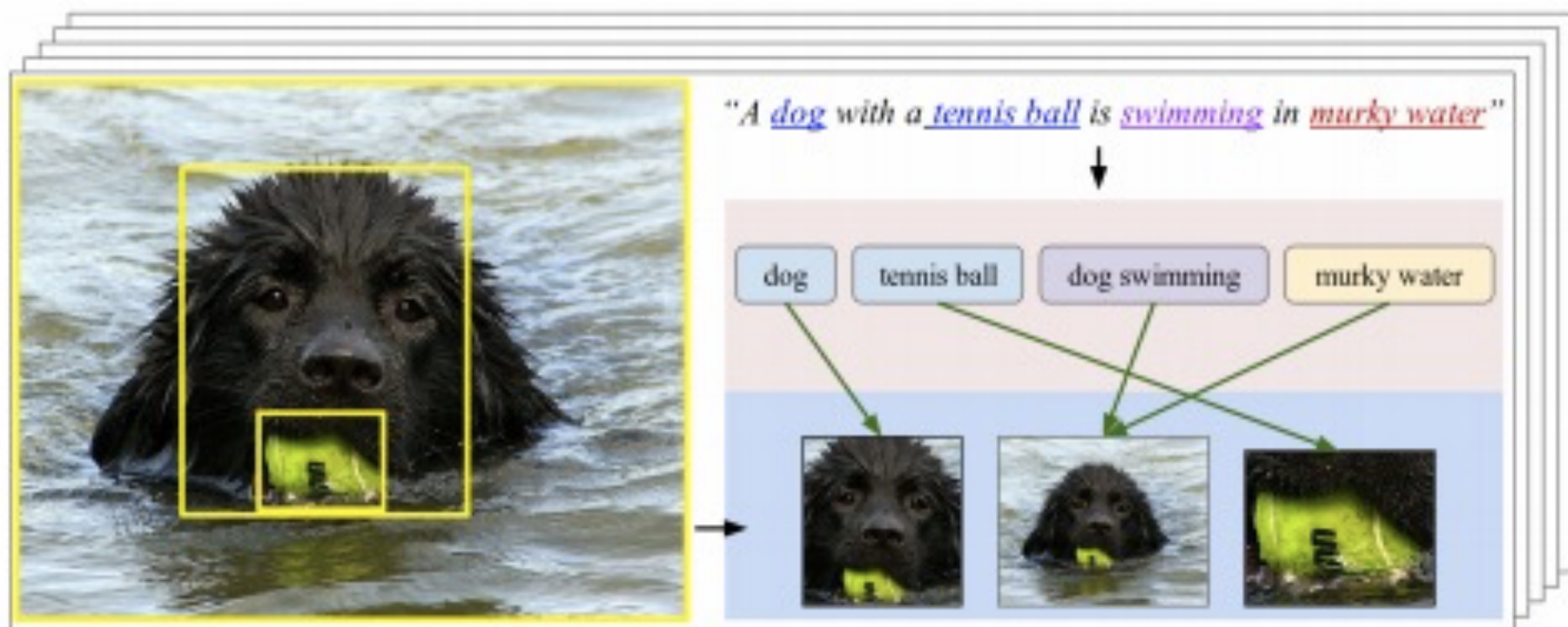


訓練用データは、画像と説明文のペアとして与えられる。
このデータでは、次の画像と、“A dog with tennis ball is swimming in murky water” という説明文が与えられている。



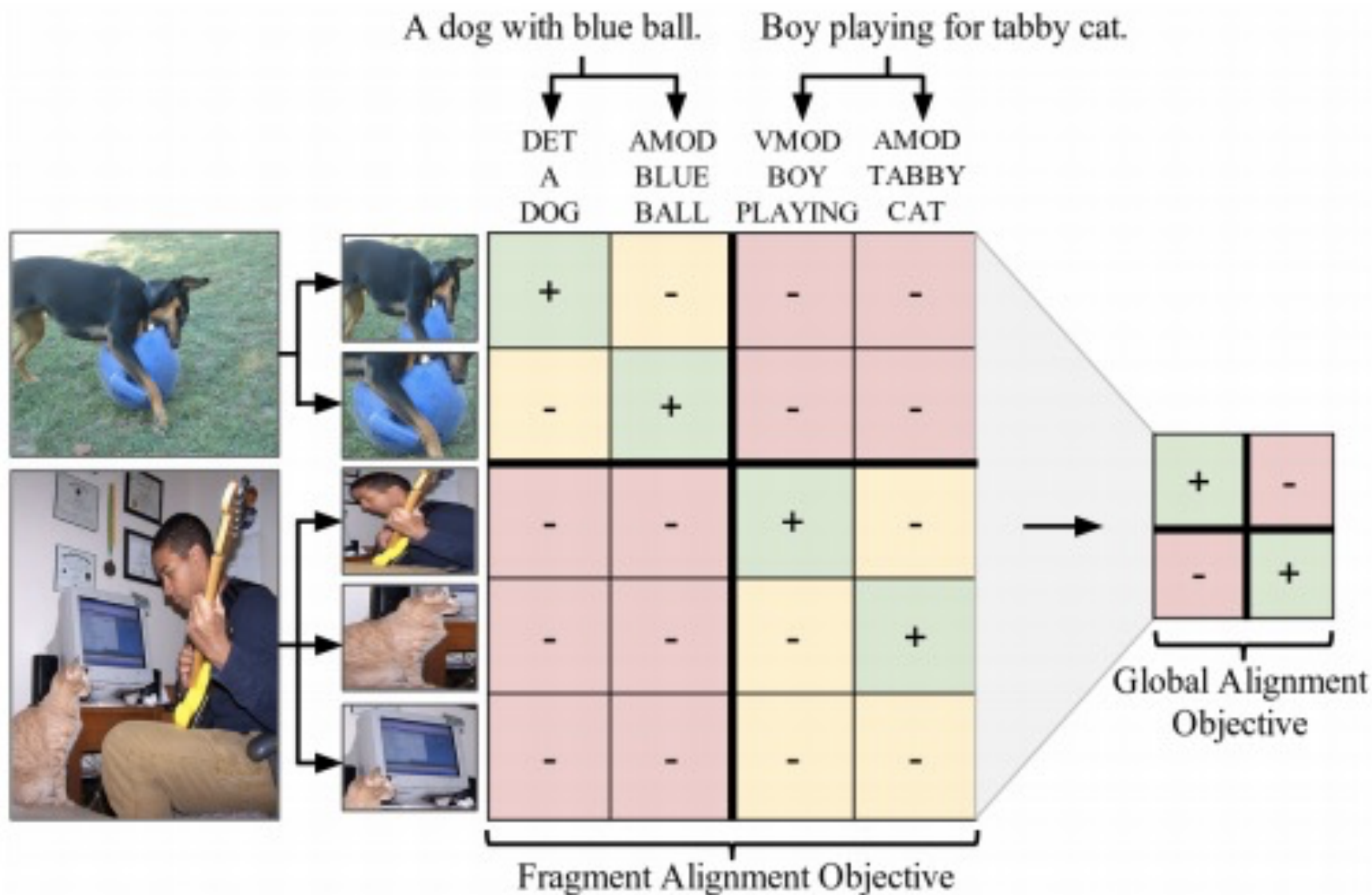
この説明文を、次に見るMarie-Catherine de Marneffeのツールで文法構造を解析して、文をフレーズに分ける。

こうして文法的な解析の結果で得られた文の一部を、今度は、画像の中で検出する。下の図の右側は、それを表している。



こうした訓練を通じて、画像の一部と、文章の一部の対応が、学習されていく。訓練が終わると、その対応表ができている。

画像の一部(横軸)と、文章の一部(縦軸)の対応表



このアプローチのもつ制限

- 我々のモデルに、は、複数の制限を免れ得ない。
モデリングという観点から見れば、文法的な従属木の解析結果の利用は、単純なものだが、いつも適切とは限らない。第一に、単一の視覚的実体を記述している単一の複合節は、複数の文の断片に分裂することがある。例えば、“black and white dog” は、二つの関係 (CONJ, black, white) と (AMOD, white, dog) とに構文解析される。逆に、画像中では、明確な根拠を持たない、従属的な関係が、多数存在する。(例えば、“each other” のように。)

このアプローチのもつ制限

- さらに、“three children playing”のようなフレーズは、ある特定の数の視覚的実体を記述しているのだが、モデル化されない。一方で、我々は、単語や二つの語の組み合わせより、こうした(文法的)関係が、より強力な表現を生み出すことを示してきた。文の断片の、より注意深い扱いが、さらなる改良へと、我々を導くだろう。
- 画像の側で言えば、RCNNの非最大制限(non-maximum suppression)が、時として、例えば、一人の人間の中に、複数の人間を検出することがある。このモデルは、検出については、いかなる空間的情報も考慮していないので、二人の異なる人間と一人の人間の過剰な検出のあいだの曖昧さをなくすことは、このモデルには難しい。

Karpathyのこれらの指摘は、極めて重要である。

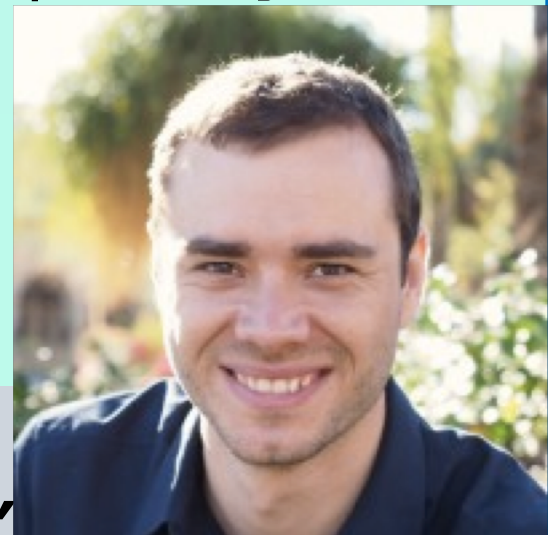
画像記述を生成するビジュアルと意味との対応付け

**"Deep Visual-Semantic Alignments
for Generating Image Descriptions"**

Andrej Karpathy, Li Fei-Fei

<http://goo.gl/efGBgN>

2015年



bouquet of
red flowers

tablet

bottle of water

glass of water with
ice and lemon

cup of coffee

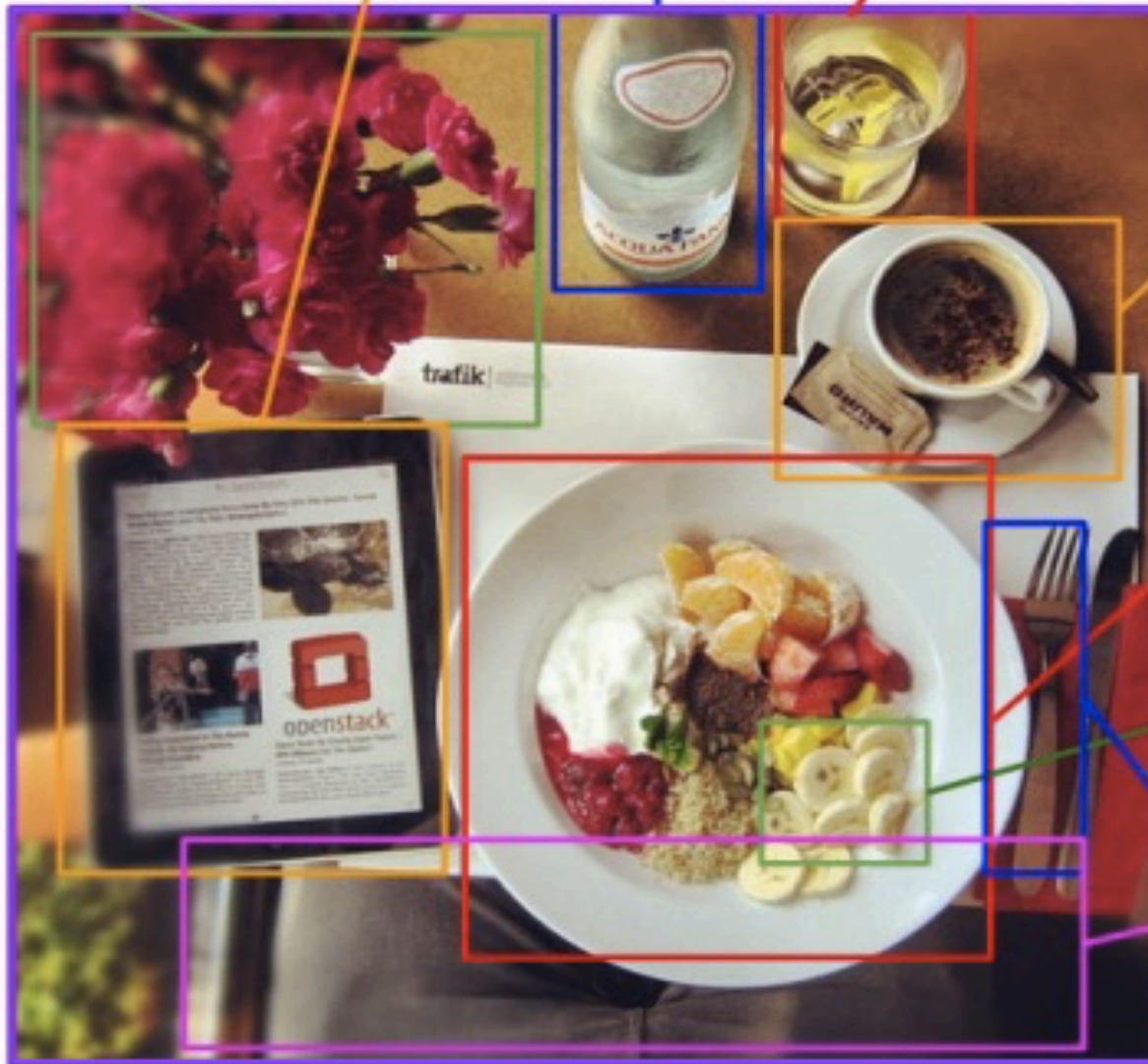
dining table
with breakfast
items

plate of fruit

banana
slices

fork

a person
sitting at a
table



我々のモデルは、言語を豊かなラベル空間として扱って、
イメージ領域の記述を生成する。

参照と対応関係

- 我々の基本的な洞察は、人間によって書かれた文章は、画像の中の特定の、ただ知られていない位置を参照しているということである。
 - 我々は、この隠れた対応関係を推論したいと思う。結果的な目標は、画像領域からこれらの小さな断片を生成することを学習することである。
 - 我々は、オブジェクトをランク付けして、画像領域を従属関係の木構造に対応付ける、Karpathyらのアプローチの上に、システムを構築した。
-

文の句構造を解析して、 型付けられた従属性解析を生成する

**"Generating Typed Dependency Parses from
Phrase Structure Parses"**

Marie-Catherine de Marneffe,
D. Manning et al.

<http://goo.gl/wSSwBR>

2006年



Visual Semantics と文の意味

語の意味から、文の意味へ

- 画像のcaption付け周辺の取り組みは、自然言語の機械処理にとって、とても興味ふかい問題を提起している。ここでは、「語の意味」ではなく「文の意味」が、また、文の「文法的構造」が、ニューラル・ネットワークの文脈では初めて、直接的・間接的に取り上げられているからである。
- もちろん、caption付けの基本になっているのは、画像中のオブジェクトの抽出とその分類である。それは、画像の一部と「語」の対応付けである。同時に、それは、文の一部としての「語」と、画像の対応付けでもある。
- 複数の画像から一つの画像が構成され、複数の語から一つの文が構成されているという、対応関係がある。

文から、現実の対応物を見つける

- Karpathy(+ de Marneffe) の試みは、いわば、文からその現実の(この場合は画像だが)、対応物を見つけようとするものである。そのために、**文を文法的に解析することから始める。これは、正しいアプローチだと思う。**
- このアプローチのいくつかの技術的な問題は、既に見たように、Karpathy自身が気付いている。これは、基本的には、文法の問題である。
- ただ、自然言語化を、知覚可能な実在する対象にかぎったとしても、静止画像からは、現実の事物の運動を正確には把握できない(ある意味、「無い物ねだり」なのだが)。
- **Scene graphを言語化すると、結局は、「...がある。」という表現に帰着する。ただ、自然言語の文の主役は、様々な運動を表す「動詞」なのである。**

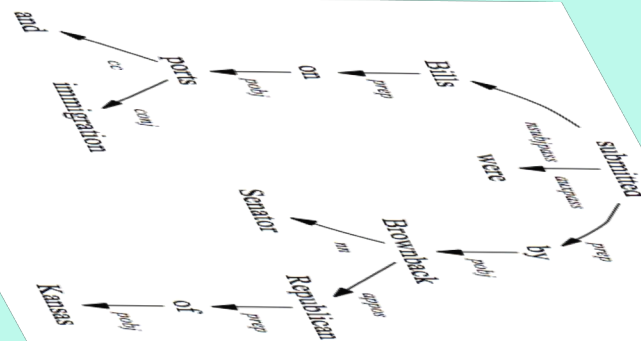
文から、現実の対応物を見つける

- ただ、もっと大きな問題もある。文が、それゆえ文法が対象としているのは、現実の「もの」だけではない。de Marneffe自身があげているサンプル “Bills on ports and immigration were submitted by Senator Brownback, Republican of Kansas.” は、現実起きたことに言及しているのだが、もの中心の素朴な対応理論では、その対応物を見つけるのは困難である。
- 現実ではないものも、当然、言語で表現可能である。
「我輩は猫である。名前はまだない」に、対応するものは？

Karpathy + de Marneffe

"A dog with tennis ball is swimming in murky water."

文の世界



文法構造

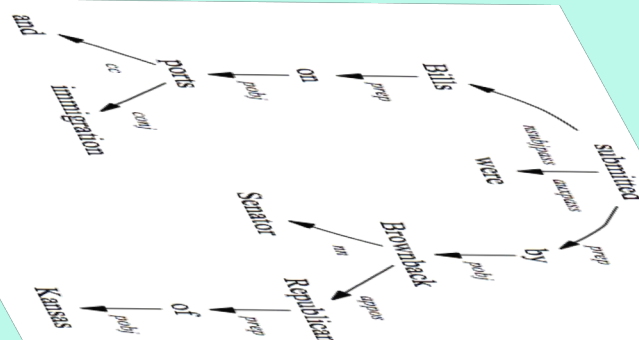


画像の世界

Karpathy + de Marneffe

"Bills on ports and immigration
were submitted by Senator
Brownback, Republican of
Kansas."

文の世界



文法構造

?

現実の世界

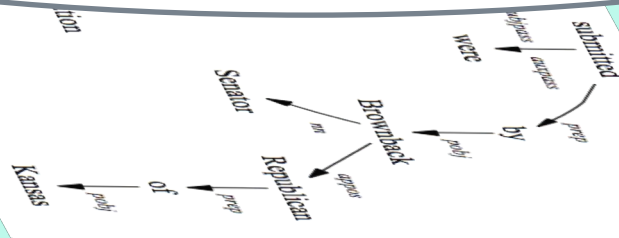
あらためて、「文の意味」を考える

- その意味では、Chomskyのいう、意味の「参照主義」「外部論者(externalist)」批判は、興味ふかい。
- ただ、画像へのcaption付けのようなアプローチが、不要だとは思わない。それは、事物の記述という範囲の中だとしても(願うらくは、それが事物の運動の記述にまで拡大することを期待しているのだが)、文の意味と文の構造について、何かを見つけ出していくであろう。
- おそらくは、意味の世界は、現実の世界とではなく、文の生み出す世界と対応することになるのだと思う。逆に言えば、人間の言語能力こそが、現実世界の把握を含めて、意味の世界を独自のスタイルで作り出しているのである。

"Bills on ports and immigration
were submitted by Senator
Brownback, Republican of Kansas."

文の世界

文法構造



意味の世界

現実の世界



意味の表現をめぐる様々な試み II

-- 意味の論理的な表現の試み --

Minimal Recursion Semantics
Deep parsing in Watson

Minimal Recursion Semantics: An Introduction

ANN COPESTAKE, DAN FLICKINGER, CARL POLLARD
and IVAN A. SAG

2005年

[http://lingo.stanford.edu/sag/papers/copestake.p
df](http://lingo.stanford.edu/sag/papers/copestake.pdf)

Main general criteria for computational semantics

- ❑ **Expressive Adequacy.**

The framework must allow linguistic meanings to be expressed correctly.

- ❑ **Grammatical Compatibility.**

Semantic representations must be linked cleanly to other kinds of grammatical information (most notably syntax).

- ❑ **Computational Tractability.**

It must be possible to process meanings and to check semantic equivalence efficiently and to express relationships between semantic representations straightforwardly.

Main general criteria for computational semantics

□ **Underspecifiability.**

Semantic representations should allow underspecification (leaving semantic distinctions unresolved), in such a way as to allow flexible, monotonic resolution of such partial semantic representations.

The first and second criteria are the object of much ongoing work in semantics, only a small subset of which claims to be computational in its aims. But computational linguists have to balance these requirements with those imposed by the third and fourth criteria.

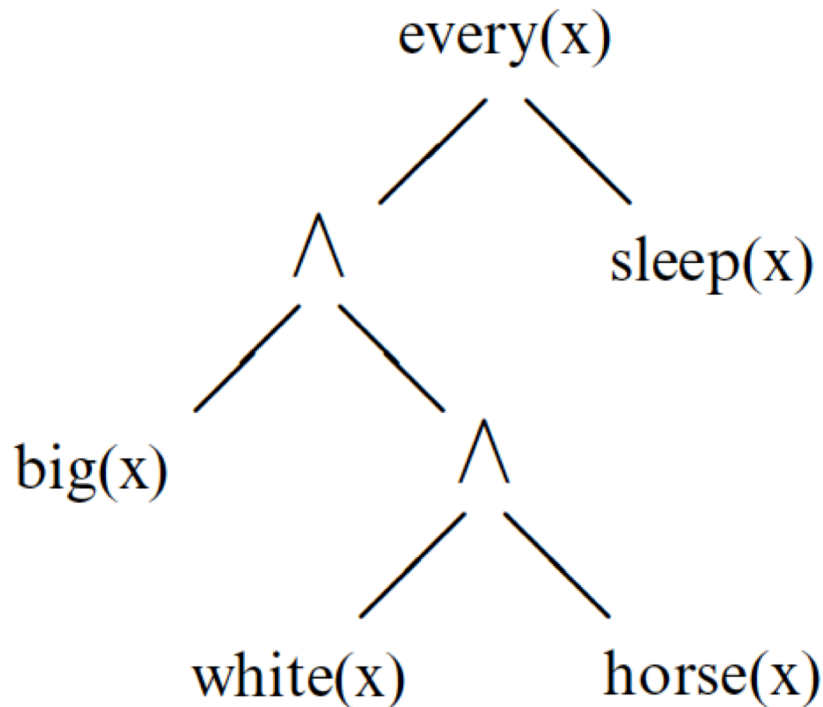
MRS Representation

“Every big white horse sleep”

a. Every big white horse sleeps.

b. $\text{every}(x, \bigwedge(\text{big}(x), \bigwedge(\text{white}(x), \text{horse}(x))), \text{sleep}(x))$

c.

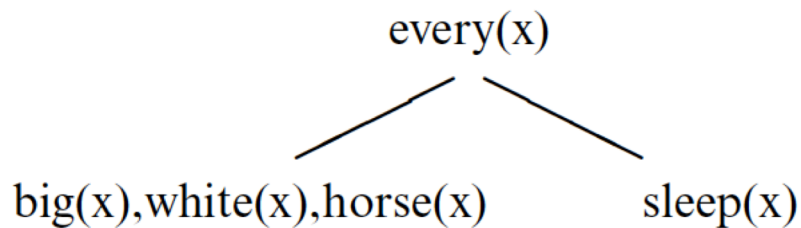
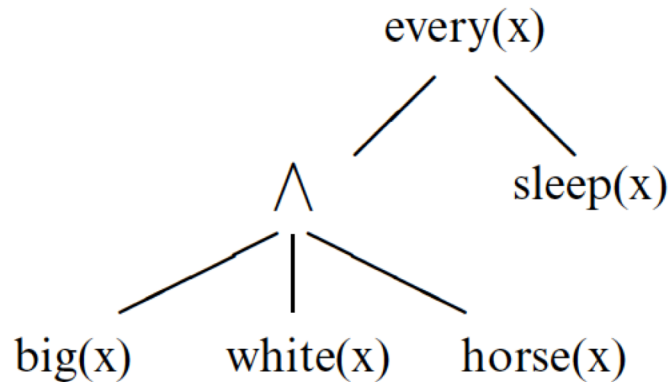


MRS Representation

“Every big white horse sleep”

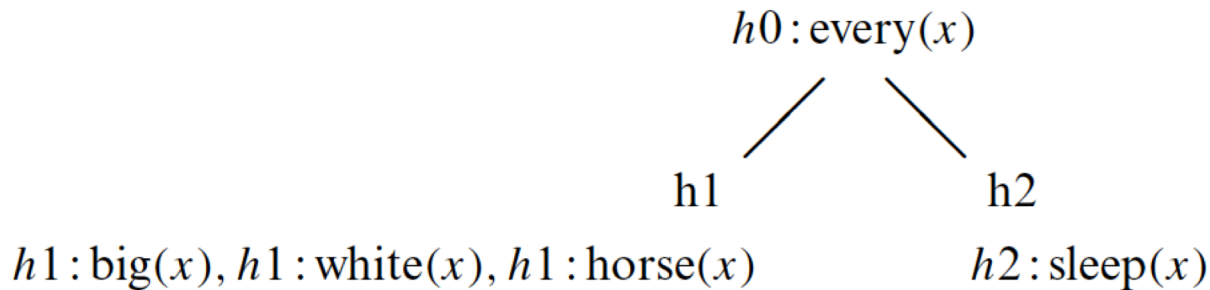
a. $\text{every}(x, \wedge(\text{big}(x), \text{white}(x), \text{horse}(x)), \text{sleep}(x))$

b.



Flat representation

“Every big white horse sleep”



$h0: \text{every}(x, h1, h2), h1: \text{big}(x), h1: \text{white}(x), h1: \text{horse}(x), h2: \text{sleep}(x)$

$h1: \text{white}(x), h0: \text{every}(x, h1, h2), h1: \text{big}(x), h2: \text{sleep}(x), h1: \text{horse}(x)$

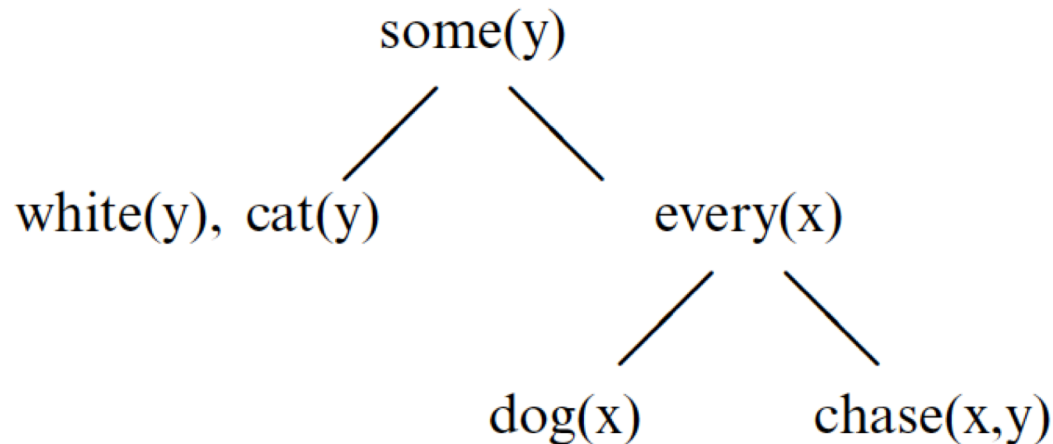
$h0: \text{white}(x), h1: \text{every}(x, h0, h3), h0: \text{big}(x), h3: \text{sleep}(x), h0: \text{horse}(x)$

MRS Representation

“Every dog chases some white cat”

a. $\text{some}(y, \text{white}(y) \wedge \text{cat}(y), \text{every}(x, \text{dog}(x), \text{chase}(x, y)))$

b.



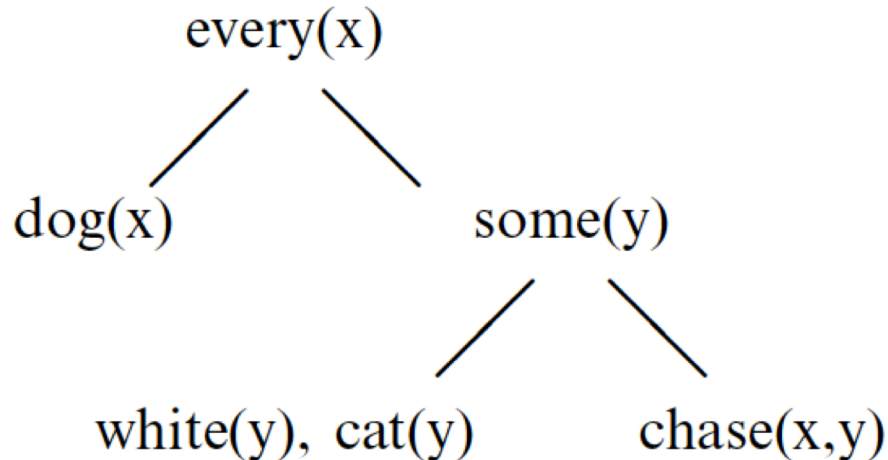
c. $h1: \text{every}(x, h3, h4), h3: \text{dog}(x), h7: \text{white}(y), h7: \text{cat}(y),$
 $h5: \text{some}(y, h7, h1), h4: \text{chase}(x, y)$

MRS Representation

“Every dog chases some white cat”

a. $\text{every}(x, \text{dog}(x), \text{some}(y, \text{white}(y) \wedge \text{cat}(y), \text{chase}(x, y)))$

b.



c. $h1: \text{every}(x, h3, h5), h3: \text{dog}(x), h7: \text{white}(y), h7: \text{cat}(y), |$
 $h5: \text{some}(y, h7, h4), h4: \text{chase}(x, y)$

An elementary predication
contains exactly four components

1. a handle which is the label of the ep
2. a relation
3. a list of zero or more ordinary variable arguments of the relation
4. a list of zero or more handles corresponding to scopal arguments of the relation (i.e., holes)

Future Structure (略)

“every dog probably sleeps”

$$\left[\begin{array}{l} \text{mrs} \\ \text{HOOK} \left[\begin{array}{l} \text{hook} \\ \text{GTOP } \boxed{1} \quad \text{handle} \\ \text{LTOP } \boxed{7} \quad \text{handle} \end{array} \right] \\ \text{RELS} < \left[\begin{array}{l} \text{_every_rel} \\ \text{LBL } \boxed{2} \quad \text{handle} \\ \text{ARG0 } \boxed{3} \quad \text{ref-ind} \\ \text{RSTR } \boxed{4} \quad \text{handle} \\ \text{BODY } \text{handle} \end{array} \right], \left[\begin{array}{l} \text{_dog_rel} \\ \text{LBL } \boxed{6} \quad \text{handle} \\ \text{ARG0 } \boxed{3} \end{array} \right], \left[\begin{array}{l} \text{_prbly_rel} \\ \text{LBL } \boxed{7} \quad \text{handle} \\ \text{ARG1 } \boxed{8} \end{array} \right], \left[\begin{array}{l} \text{_sleep_rel} \\ \text{LBL } \boxed{9} \\ \text{ARG1 } \boxed{3} \end{array} \right] > \\ \text{HCONS} < \left[\begin{array}{l} \text{qeq} \\ \text{HARG } \boxed{1} \\ \text{LARG } \boxed{7} \end{array} \right], \left[\begin{array}{l} \text{qeq} \\ \text{HARG } \boxed{4} \\ \text{LARG } \boxed{6} \end{array} \right], \left[\begin{array}{l} \text{qeq} \\ \text{HARG } \boxed{8} \\ \text{LARG } \boxed{9} \end{array} \right] > \end{array} \right]$$

$\langle h1, h7, \{h2: \text{every}(x, h4, h5), h6: \text{dog}(x), h7: \text{prbly}(h8), h9: \text{sleep}(x)\},$

$\{h1 =_q h7, h4 =_q h6, h8 =_q h9\}\rangle$

Introduction to Grammar Engineering with HPSG

Yi Zhang

2008年

<http://www.coli.uni-saarland.de/~yzhang/ge-ss08/lecture-05.pdf>

HPSG: Head-driven phrase structure grammar

From Syntax to Semantics

- Syntactic analysis helps to reveal the underlying meaning of texts
- Need formalism to represent all semantic distinctions which are syntactically (or morphologically) marked
 - Need to underspecify semantic distinctions which don't correspond to differences in form
 - Abstract away from non-semantic information (case, word order, etc.)
 - Language independence

Deep parsing in Watson

M.C.McCord, J.W.Murdock, B K.Boguraev

http://brenocon.com/watson_special_issue/03%20Deep%20parsing.pdf

Watsonは、自然言語を解析する能力を持つ

- Watsonと、Alexa, Google Homeといった「パーソナル・アシスタント・システム」との最大の違いは、文の解析能力の違いにある。
- Watsonは、浅いレベルでは、文を文法的に構文解析して、その上で、深いレベルで、その論理的な意味を分析しようとする。自然言語解析の枠組みは、ESG(English Slot Grammar)という文法理論に基づいている。
- Watsonは、この言語解析の能力を利用して、自然言語で記述されたWikipedia等の膨大なドキュメントから、形式化された知識データベースを作成することができる。
- ただ、Watsonは、こうした解析から得られたデータから、論理的な推論ルールに基づいて、答えを探しているわけではない。

Slot Grammarの基本文献

□ 次の三つの論文が、Slot Grammarの基本的文献である。

- “A Formal System for Slot Grammar”
<http://goo.gl/KR56Es>
 - “The Slot Grammar Lexical Formalism”
<http://goo.gl/57vxL4>
 - “Using Slot Grammar”
<https://goo.gl/P1ZgcQ>
-

Slotとは？

- Slotというのは、基本的には、文中の句の**文法的な役割**を表している。いくつかのSlotの例を挙げる。

subj	subject (主語)
obj	direct object (直接目的語)
iobj	indirect object (間接目的語)
comp	predicate complement (述語補語)
objprep	object of preposition (前置詞目的語)
ndet	NP determiner (名詞句限定子)

- Slotのもう一つの役割は、**意味論的な役割**である。
例えば、compは、述語の「位置を持つ引数」として、言葉の意味を表す。
-

Slot Grammarでの文法的解析

- 次の図は、“Mary gave John a book”を、Slot Grammarで解析したものである。
- 文法的には、スロット subjは、“Mary”で「満たされ」、スロット iobjは、“John”で「満たされ」ている。等々。こうした観点からは、スロットが文法的な役割を果たしていることがわかる。

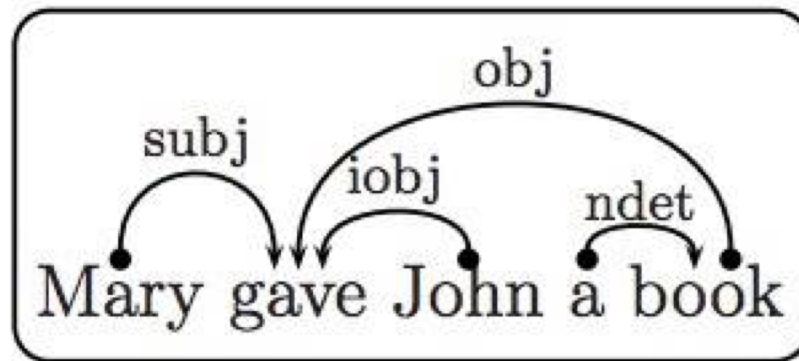
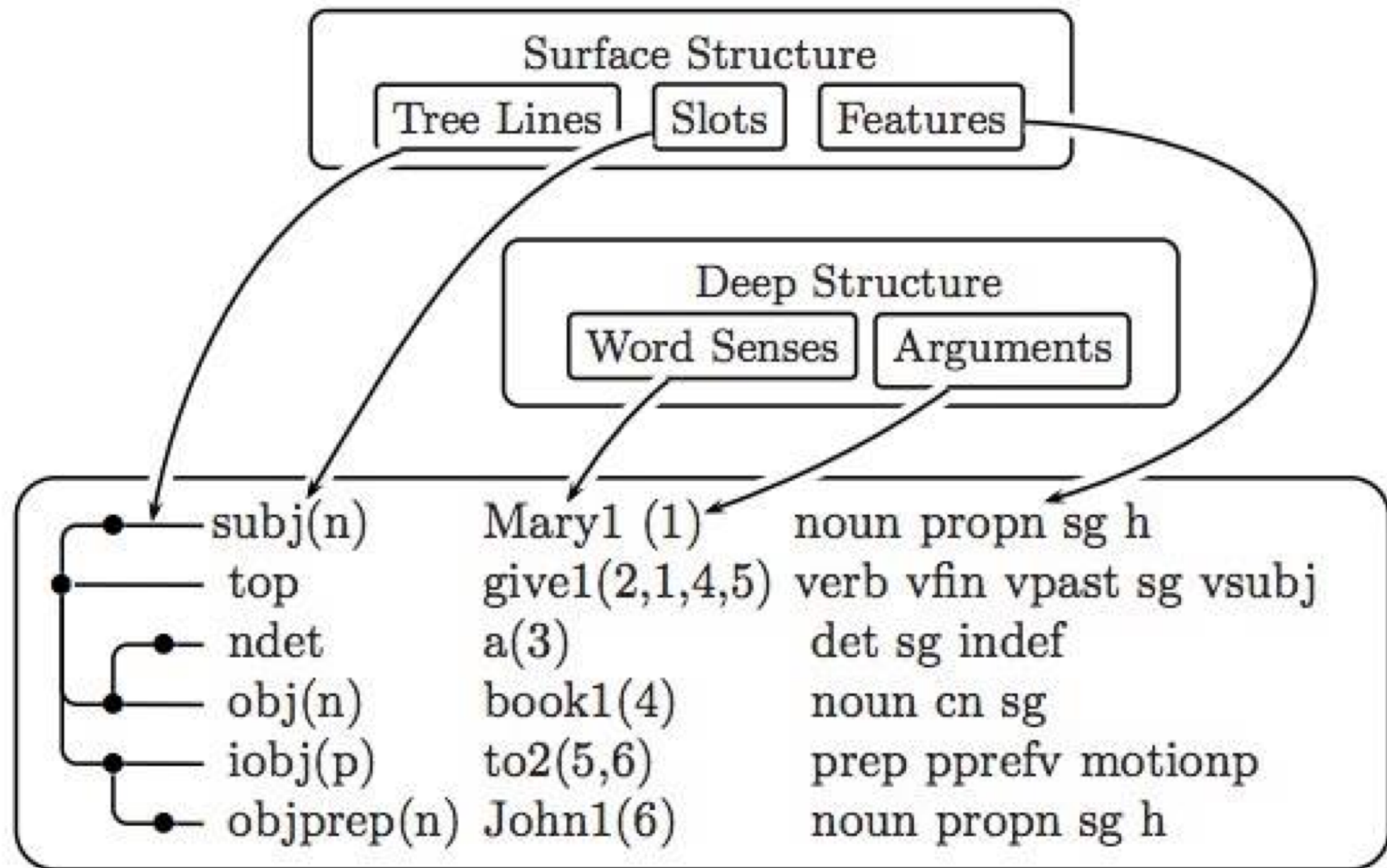


Figure 1: Slot filling for *Mary gave John a book*

“Mary gave a book to John.” の解析

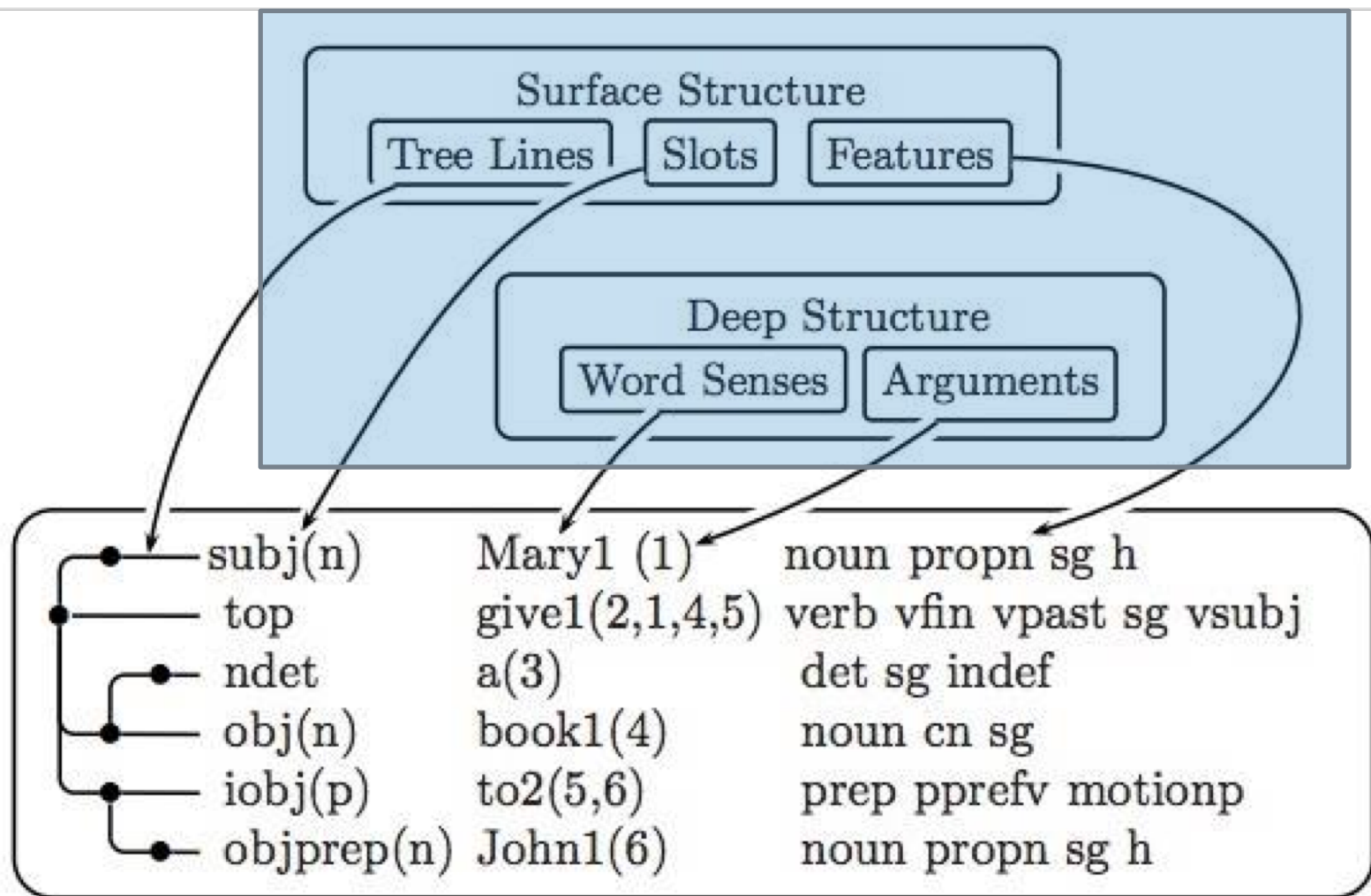


「表層構造 (Surface Structure)」と「深層構造 (Deep Structure)」

- 先の図で重要なのは、文法的な構造を表す「表層構造 (Surface Structure)」と意味を担う「深層構造 (Deep Structure)」の二つから文を解析しようとしているところだ。(もっとも、これは50年以上前のChomsky以来の古典的な観点だ)
 - もう少し、詳しく見てみよう。
Slot Grammarで「表層構造 (Surface Structure)」を担っているのは、"Tree Line"と"Slot"と"Features"だとされている。一方で、「深層構造 (Deep Structure)」を担っているのは、"Word Senses"と"Arguments"だとされる。
-

“Mary gave a book to John.”

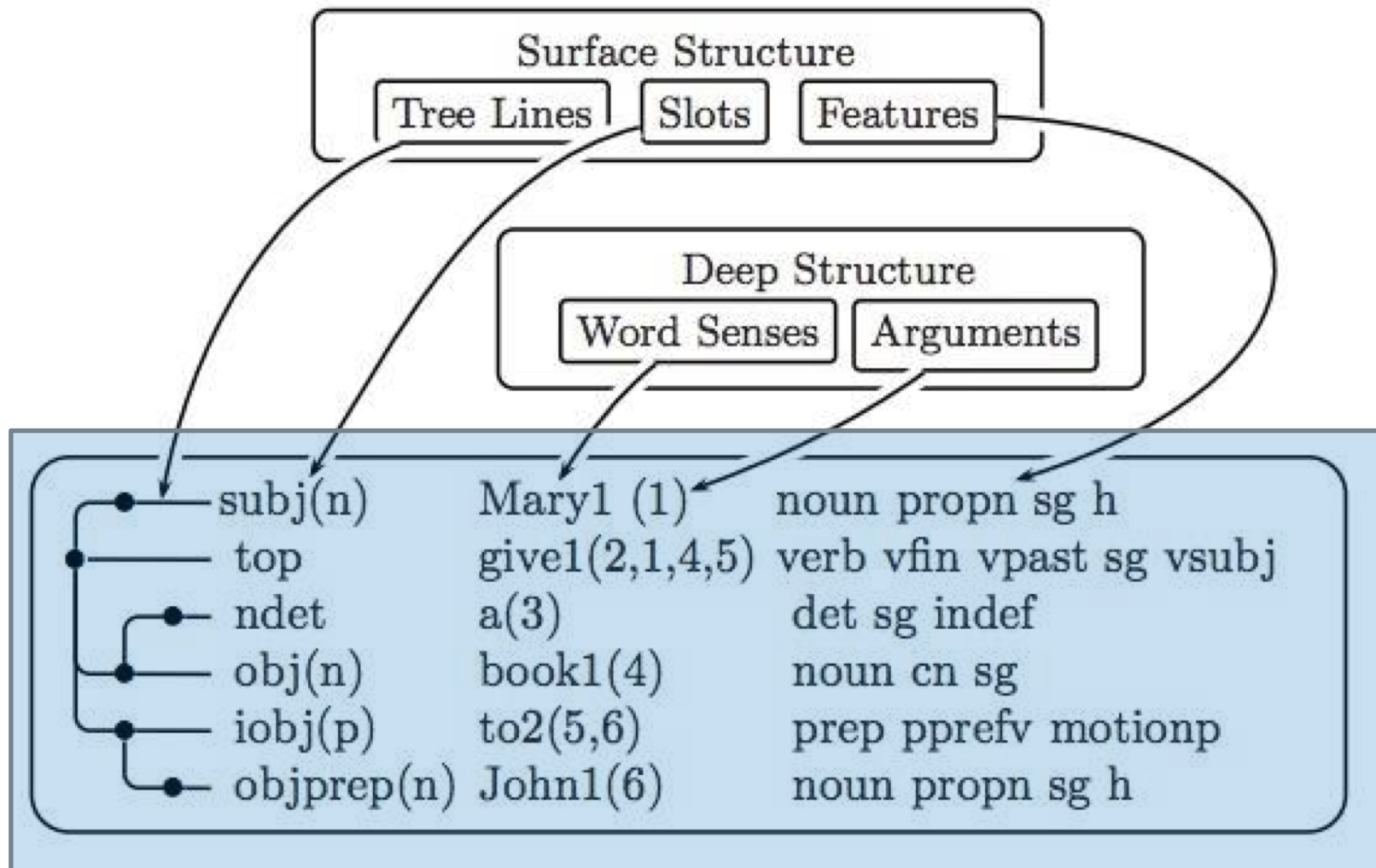
表層構造・深層構造



- 今度は、この図の中心をなす下半分に注目しよう。
この領域は、大きく、三つの部分からなる。
- 一番左は、Slot単位の構文解析木 Tree Line である。これは、文法的な「表層構造 (Surface Structure)」を表している。
- 中央部分は、語義 Word Sense だが、それぞれは「原形」としてLexiconの「見出し語」に対応している。Lexiconの「意味フレーム (Sense Frame)」は、「見出し語」に対して、Slot を「引数」として対応づける情報を与える。ただ、ここで第一引数として与えられているのは、それぞれの語の文中での出現順位である。
- 結局、以前に見たように、次の論理式が、この文の「意味」を表現することになる。
$$\exists e \exists y (\text{book}(y) \wedge \text{give1}(e, \text{Mary}, y \text{ John}))$$
- この図の右側は、それぞれの語のFeatureを表している。Featureは、Lexiconの「意味フレーム (sense Frame)」で与えられる。

“Mary gave a book to John.”

Tree Line + Word Sense + Features



Featureサンプル

- Mary のFeatureは、“noun propn sg h”だが、それぞれのFeatureの意味は、

noun: 「名詞」

propn: proper noun で「固有名詞」

sg: single で「単数」

h: humanで、「人間」

いずれも、辞書には出てきそうな情報である。

- giveのFeatureは、“verb vfin vpast sg vsubj” だが、それぞれのFeatureの意味は、

verb: 「動詞」

vfin: finite verb で「定動詞」

vpast: 「動詞過去形」

sg: 「単数」 である。

Slot Grammarでの意味的解析

- 意味論的な観点からスロットの役割を見るには、giveの語義を考えよう。それを論理的に表現したものを give1とすると、**give1(e, x, y, z)**は、「イベントeで、xはyにzを与える」を意味する。
- この時、先の図の文章の論理的な表現は、
 $\exists e \exists y (\text{book}(y) \wedge \text{give1}(e, \text{Mary}, y, \text{John}))$
になる。
- こうした論理的な観点からは、スロット subj, obj, iobj は、述語**give1**の引数 x, y, z の名前とみなすことができる。こう考えてもいい。これらのスロットは、動詞の意味を表す述語の位置を持つ引数だと。

Complement Slotと Adjunct Slotと

- このように述語の引数になるスロットは、complement Slot と呼ばれ、Slot Grammar Lexicon(スロット文法語彙集)という辞書を通じて、語義に関連づけられる。
 - 一方、それ以外のスロット、ndet(冠詞・定冠詞)のようなスロットは、adjunct Slotと呼ばれて、Slot Grammarのsyntaxに関連することになる。
-

Slot Grammar Lexicon

- Slot Grammarで重要な役割を担っているのは、その辞書、Lexicon 語彙集である。ここでは、次の論文に従って、その形式 SLF(Slot Grammar Lexical Formalism)を紹介しよう。"The Slot Grammar Lexical Formalism"<http://goo.gl/57vxL4>
- まず、もっとも簡単な形式は、次のようになる。これは、“access” という語の登録形式である。

```
access  
  < v obj  
  < n (p to)
```


Slot Grammar Lexicon

- ここで、見出語は、accessである。この見出語について二つの要素 v obj と n (p to) が登録されている。
vはverbで「動詞」を表し、nはnounで「名詞」を表している。これらは、見出語の「品詞」にあたる。この表現は、accessは、動詞と名詞があることを示している。

access

< v obj

< n (p to)

見出語

v は動詞

n は名刺

- 品詞 v, n のあとに続く obj, (p n)は、スロットを表している。

Slot Grammar Lexicon

□ 次のエントリー

give < v obj iobj

は、giveは動詞(v)で、二つのスロット obj と iobj を取ることを表している。

Alice gave the book to Bob.

Alice gave Bob the book.

Alice gave the book to Bob.
 obj iobj

Alice gave Bob the book.
 iobj obj

- 最初の文では、スロット obj は、名詞句(NP) the book で満たされ、スロット iobj は前置詞句(PP) to Bobで満たされている。二つ目の文では、スロット iobj は、異なる位置で、名詞句 Bobで満たされ、スロット obj は、やはり異なる位置で、名詞句 the bookで満たされている。
- 二つの文で、iobjを満たしているものは、NPとto-PPで異なるのだが、満たしているものの論理的役割は同じなので、二つの文、どちらでも、Bobが本を与えられた人物であることがわかる。

Alice gave the book to Bob.
 obj iobj

Alice gave Bob the book.
 iobj obj

Lexconのエントリーの例

admonish < v obj (comp (p about against for))

admonish < v obj (p about against for)

admonition < n obj (p about against for)

drive < obj (comp lo)

She drove her car out .
 obj (comp lo)

She drove her car onto the interstate .
 obj (comp lo)

She drove her car fifty miles .
 obj (comp lo)

意味フレーム

- Slot GrammarのLexiconでは、
見出語 < 品詞 スロット の後ろに、その語の「特徴」を表す情報が並ぶ。
- その語についてのその他の付加的な情報も、この後ろに並ぶ。Slot Grammar Lexiconでは、こうした記述をSense Frame「意味フレーム」と呼んでいる。

見出語 < 品詞 スロット 諸特徴 その他の情報

Features

見出語 < 品詞 スロット 諸特徴 その他の情報

- 「諸特徴 (Features)」と簡単に書いたが、品詞ごとに、動詞についてのFutures、名詞についてのFutures、形容詞についてのFeatures、副詞についてのFuturesと、数多くのFeaturesがある。
- 先の論文、“The Slot Grammar Lexical Formalism” <http://goo.gl/57vxL4> の多くの部分は、このFeaturesの記述に当てられている。
- Featuresの詳細については、スペースの関係で、ここでは触れない。次に、その一部のリストを示す。

Part II

構成的意味理解への予備的アプローチ



Part II 構成的意味理解への予備的アプローチ

Agenda

□ Minimalist Programと Categorical Grammar

□ Lambek vs Lambek

Categorical GrammarとPregroup Grammar

■ The mathematics of sentence structure

■ From word to sentence

□ 理論とモデル

■ 意味の二項性

■ 「ルール」の世界と「たとえ」の世界

■ 語の意味のモデル – 素朴な対応理論

■ 理論とモデル – 対応の形式化

■ 理論とモデル 2 – 数学でのモデルの利用

■ Category論とFunctor Semantic

文法の新しい定式化と 認識へのモデル論的アプローチ

- この第二部は、文の構成性と意味の構成性の双方の性質をあわせ持つ言語理論を紹介する第三部の理論的フレームワークの予備的導入である。
- 第一に、文法理論の新しい定式化として、Lambekの Pregroup Grammarが紹介される。筆者はそれを、言語学でのチョムスキーのMinimalist Programと、同じ内容、同じ指向性を持つものとして捉えている。
- 第二に、意味の理論、認識の理論にとって、本質的に重要な役割を果たすと筆者が考えている、認識へのモデル論的アプローチが紹介される。LawvereのFunctor Semanticsは、第三部のDisCO意味論の直接の基礎である。

基本的な観察

文と意味の「構成性(compositionality)」

- 文は文法に従って語から構成される
- 文の意味は、語の意味に依存する
 - 文法的に正しいが意味がない文もある
Colorless green ideas sleep furiously.
- 文の意味は、文の文法的構成に依存する
 - 二つの意味を持つ、あいまいな文
I saw a man with a telescope.

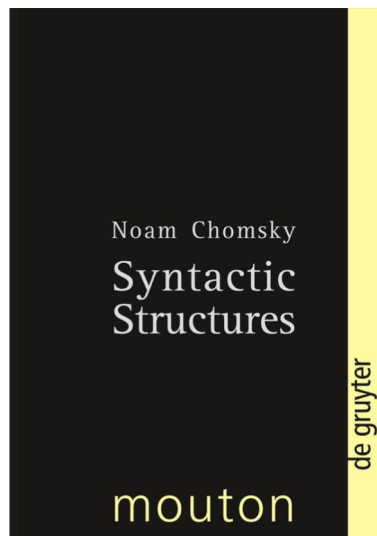
意味の「同一性」 / 意味の共通表現の存在

- 言語が異なっても、「同じ」意味を持つ語が存在する。
- 言語が異なっても、「同じ」意味を持つ文が存在する。

Minimalist Programと Categorial Grammar



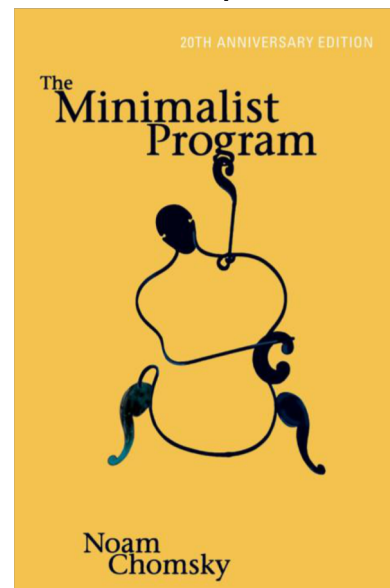
1957年



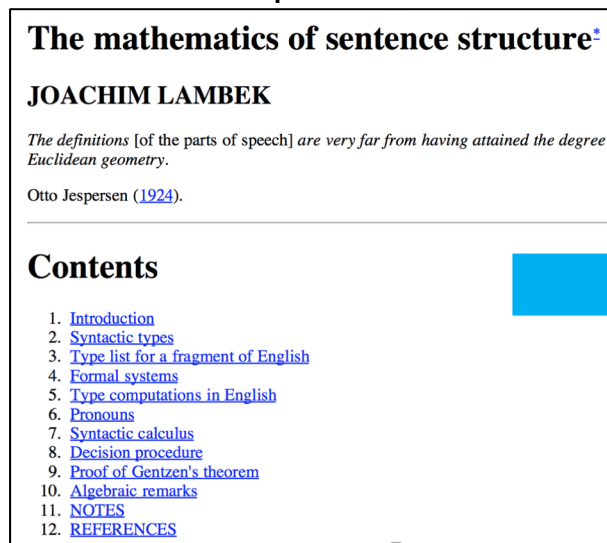
Chomsky

48年

1995年



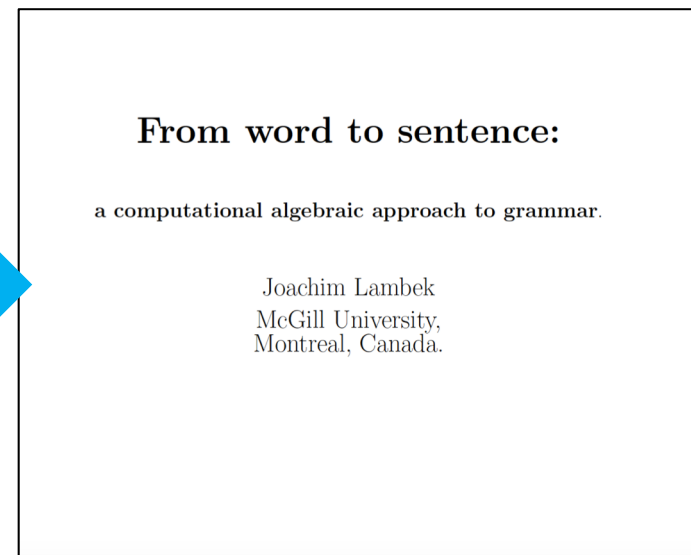
1958年



Lambek

50年 !

2008年



Minimalist文法とカテゴリー文法の収斂について

1 AT THE CARTESIAN WELL: THE MINIMALIST PROGRAM & CATEGORIAL GRAMMAR

Imagine the following scene. You are at your favorite beer hall somewhere in Amsterdam—let's call it the Cartesian Well. Well known meeting place of intelligentsia, you are not surprised when a thin person dressed all in black sidles up to you and whispers in your ear, "Have I got a linguistic theory for you!" You of course yawn, have heard many such fables in your time; besides you have drunk too much. "No, wait," the figure grabs your shoulder, "I've discovered that Chomsky's latest approach to syntax and categorial grammar are *converging*."

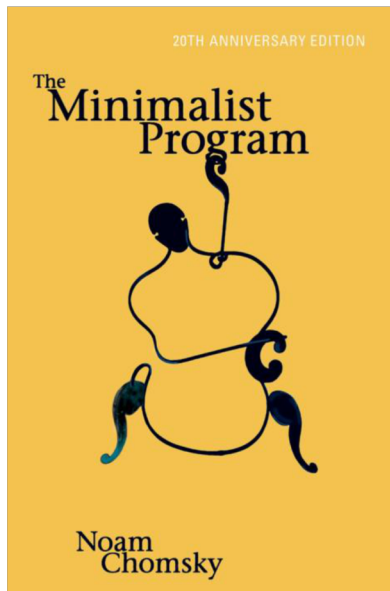
Another flat-earth cultist? you think. "Well, hear me out—at least let me buy you another beer." So you to listen to the tale:

4 CONCLUSIONS FROM THE BEER HALL

To summarize, given the current push towards "minimalism" in the so-called government-and-binding approach seems to have eliminated both government (and binding, not discussed here) in favor of a single hierarchical concatenation operator that meshes perfectly with the classical theory of categorial grammar, as well as providing a natural *explanation* for the observed grammatical relations and a transparent framework on which to build a model of sentence processing. While this trend surely does not solve all our "religious" problems, it certainly goes a long way towards taking down the "barriers" to a mature, ecumenical framework within which to reach common ground among what has long appeared to be quite disparate accounts of natural language. Perhaps we can all now drink beer together.

1995年 Berwick & Epstein

"On the Convergence of 'Minimalist' syntax and Categorical Grammar"



1995年

On the Convergence Of 'Minimalist' Syntax and Categorial Grammar

Professor Robert C. Berwick, MIT Department of Computer Science

Professor Samuel David Epstein, Harvard Department of Linguistics
Cambridge, MA 02139

ABSTRACT

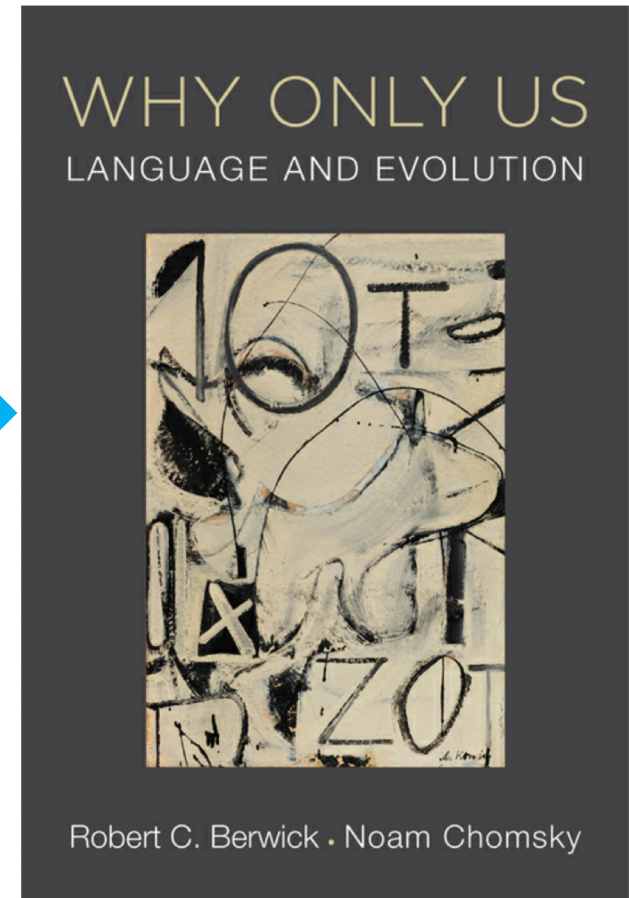
This paper shows that the so-called "Minimalist Program" of Chomsky (1993, 1995) can be given a natural interpretation as a categorial system in which there is exactly one syntactic (algebraic) operation: namely, "Hierarchically Concatenate" (HC) (what Chomsky calls "Merge"), and also replacing the representations of "D-structure," "S-structure" and transformations with the derivation lines typical of categorial systems—thus unifying two previously disparate approaches to the analysis of natural language. For example, the general "movement" rule of transformational grammar is easily seen to be a subcase of Hierarchical Concatenation of (α , β), where α is a subtree of β ; this automatically derives the usual c-command condition on so-called "empty categories." The usual semantic interpretation benefits of the categorial

interpret" sentences.

1 AT THE CARTESIAN WELL: THE MINIMALIST PROGRAM & CATEGORIAL GRAMMAR

Imagine the following scene. You are at your favorite beer hall somewhere in Amsterdam—let's call it the Cartesian Well. Well known meeting place of intelligentsia, you are not surprised when a thin person dressed all in black sidles up to you and whispers in your ear, "Have I got a linguistic theory for you!" You of course yawn, have heard many such fables in your time; besides you have drunk too much. "No, wait," the figure grabs your shoulder, "I've discovered that Chomsky's latest approach to syntax and categorial grammar are *converging*."

2016年



Berwick & Chomsky

Lambek: “Remarks on the history of categorial grammar”

- A few years later, I was invited to a symposium of the American Mathematical Society [1961] on applications of mathematics to linguistics. Hard pressed to find anything new to say, I discussed a non-associative version of the syntactic calculus.
- These ideas did not catch on with the linguistic community and even met with some antagonism by certain graduate students, **although two linguists, Noam Chomsky and Robert Lees, tried to encourage me to continue. Still, I could see the new wave of Chomsky’s generative-transformational grammar sweeping everything else aside and turned my attention to other things.**

Lambek: “Word to Sentence”

0. Preface.

- Chomsky refined and deepened his theory in subsequent years, publishing almost one book per annum. Yet, while going profoundly into the structure of natural languages, it became less accessible to computational implementation, at least in the opinion of some categorial linguists.

Prompted by a suggestion by Claudia Casadio, I proposed a new computational algebraic approach in 1998, founded in the categorial tradition and based on a slight generalization of the ubiquitous concept of a group, called a “pregroup”. In retrospect, this approach was already implicit in the work of Zellig Harris, Chomsky’s teacher, and may be viewed as a refinement of the latter.

Lambek: "Word to Sentence"

0. Preface.

- My own experience tells me that, in countries where high school students are faced with a choice between a scientific program and a humanistic one based on a study of classical languages, **future mathematicians often arise from the latter stream.**
- The principal idea is to assign to each word in the mental dictionary one or more types. (Sometimes the types of inflected forms of lexical items may be calculated by certain "metarules".) These types are elements of an algebraic system or terms of a logical system and the grammatical status of a string of words is determined by a calculation on the corresponding types.

Lambek: "Word to Sentence"

36. Postscript on Minimalism.

- Readers familiar with current linguistic theory will have noticed that many ideas underlying the present monograph have been influenced by the profound insights gained by Chomsky and his school in the last half century. However, there are technical differences in how these ideas are displayed. Mainstream (Chomskyan) grammarians assume that words are arranged in labeled bracketed strings, or equivalently as leaves on a plane tree with labeled nodes, and that sentences are generated and analyzed by manipulating these structures.

- In categorial grammars, such calculations are performed not on labeled bracketed strings of words, but on strings of types, and this is assumed to be done in real time, even as the words are uttered or perceived during actual communication. In particular, this approach rules out any movement from the end of a sentence to the beginning.
- Chomsky's ideas have undergone a number of significant changes over the years, which have usually led his many disciples to revise their methodology as well. (I am reminded of the French comic book "Asterix and Cleopatra", in which Obelix accidentally breaks off the nose of the Sphinx, whereupon the souvenir vendors hasten to hack off the noses of their replicas.)

- For the last dozen years or so, the official version of the Chomskyan enterprise has been the so-called “minimalist program” (described as “maximalist” by some). It is not clear to me which of the earlier notions have been discarded and which have been absorbed in the new program. The two texts I have looked into rely heavily on a discussion of the earlier theories. For example, Chomsky [1995] discusses government theory, binding theory, case theory, X-bar theory, phrase structure theory and theta-theory, to judge by the section headings only, not counting trace theory and optimality theory, which are mentioned elsewhere (see Lasnik et al.[2005]).

Is the reader required to study all these theories before engaging on the minimalist program?

I must confess that I haven't mastered all these theories, although I suspect that what some of them are about can be incorporated into the pregroup framework.

- Much of the recent debate among mainstream grammarians is couched in the technical language of parsing trees. There have been a number of papers analyzing the minimalist program from an algebraic point of view, for example by Stabler [1997], and by several authors in de Groote et al. [2001], showing that **minimalist grammars are weakly equivalent to multiply context-free ones**. As far as I can judge, they still rely on an arboreal representation. **While I cannot say that I have completely understood the minimalist program, I believe that in spirit, if not in technical details, pregroup grammars may be viewed as an attempt to carry out such a program.**

Lambek vs Lambek

Categorial Grammarと
Pregroup Grammar

The mathematics of sentence structure

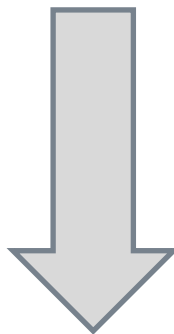
JOACHIM LAMBEK

http://lecomte.al.free.fr/ressources/PARIS8_LSL/Lambek.pdf

2. Syntactic types

$$(I) \quad (x/y)y \rightarrow x, \quad y(y \setminus x) \rightarrow x.$$

1958年の定式化



2008年の定式化

$$xx^r \rightarrow 1, \quad x^\ell x \rightarrow 1,$$

3. Type list for a fragment of English

(1) **John** **works**
 n $n \backslash s$

$n \ n \backslash s \rightarrow s$

(2) (**poor** **John**) **works**
 n/n n $n \backslash s$

$(n/n \ n) \ n \backslash s \rightarrow n \ n \backslash s \rightarrow s$

(3) (**John** **works**) **here**
 n $n \backslash s$ $s \backslash s$

$(n \ n \backslash s) \ s \backslash s \rightarrow s \ s \backslash s \rightarrow s$

(4) **John** (**never** **works**)
 n $(n \backslash s)/(n \backslash s)$ $n \backslash s$

$n \ ((n \backslash s)/(n \backslash s) \ n \backslash s) \rightarrow n \ n \backslash s \rightarrow s$

(5) (**John** **works**) (**for** **Jane**)
 n $n \backslash s$ $(s \backslash s)/n$ n

$(n \ n \backslash s) \ ((s \backslash s)/n \ n) \rightarrow s \ s \backslash s \rightarrow s$

3. Type list for a fragment of English

(6) (John works) (and (Jane rests))
 n $n \backslash s$ $(s \backslash s) / s$ n $n \backslash s$

$(n \ n \backslash s)) ((s \backslash s) / s (n \ n \backslash s))) \rightarrow s ((s \backslash s) / s s) \rightarrow s s \backslash s \rightarrow s$

(7) John (likes Jane)
 n $(n \backslash s) / n$ n

$n ((n \backslash s) / n n)$
 $\rightarrow n \ n \backslash s \rightarrow s$

(7') (John likes) Jane
 n $n \backslash (s / n)$ n

$(n \ n \backslash (s / n)) n$
 $\rightarrow s / n \ n \rightarrow s$

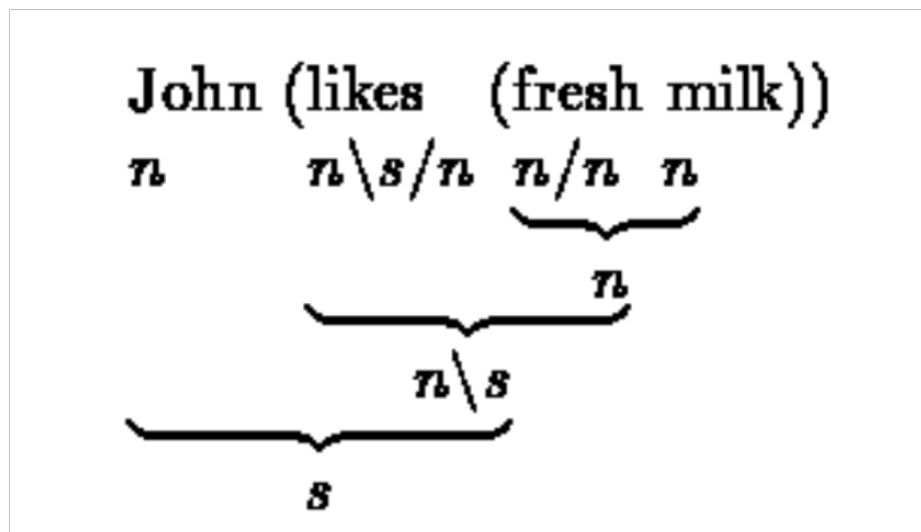
(II) $(x \backslash y) / z \iff x \backslash (y / z).$

Table I

	Word	Type	Part of Speech	Fries Class
(1)	<i>works</i>	$n \backslash s$	intransitive verb	2C
(2)	<i>poor</i>	n / n	adjective	3
(3)	<i>here</i>	$s \backslash s$	adverb	4
(4)	<i>never</i>	$n \backslash s / (n \backslash s)$	adverb	
(5)	<i>for</i>	$s \backslash s / n$	preposition	F
(6)	<i>and</i>	$s \backslash s / s$	conjunction	E, J
(7)	<i>likes</i>	$n \backslash s / n$	transitive verb	2B

カテゴリー文法の「カテゴリー」は、数学のカテゴリー論の「カテゴリー」とは、直接の関係はない。それは、従来の文法の「品詞(Part of Speech)」概念の拡大である。その拡大された「品詞」をカテゴリーと呼んでいる。このカテゴリーは、語の接続によって引き起こされる計算上の「型」を持つ。

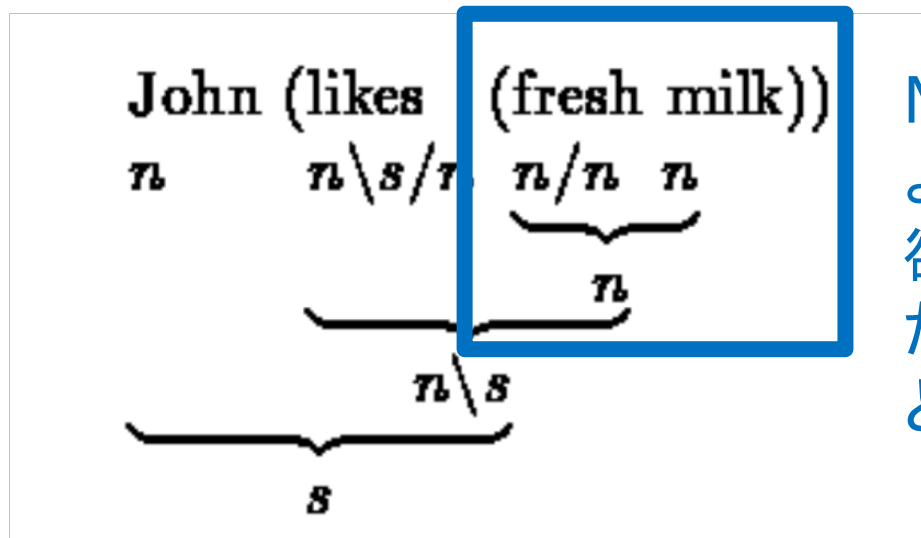
4. Formal systems



文の文法性の計算は、隣り合う語のカテゴリの型の計算から、文のカテゴリ s を導出する形式的体系で記述できる。

$$\tau_b((\tau_b \backslash s / \tau_b)((\tau_b / \tau_b)\tau_b)) \rightarrow \tau_b((\tau_b \backslash s / \tau_b)\tau_b) \rightarrow \tau_b(\tau_b \backslash s) \rightarrow s.$$

4. Formal systems



Merge (cool, water)

$\rightarrow \{water, \{cool, water\}\}$

Mergeの議論での上のような導出を思い出して欲しい。同じことだ。

ただし、ここでは "water" というラベルは不要である。

$$\tau_b((\tau_b \backslash S / \tau_b)((\tau_b / \tau_b) \tau_b)) \rightarrow \tau_b((\tau_b \backslash S / \tau_b) \tau_b) \rightarrow \tau_b(\tau_b \backslash S) \rightarrow S.$$

5. Type computations in English

- i. Insert brackets in all admissible ways.
- ii. To each word assign all types permitted by a given finite type list. (We ignore for the moment the difficulty arising from words which possess a potentially infinite number of types, as do the chameleons `and' and `only').
- iii. For each grouping and type assignment compute the type of the total expression.
- iv. Select that method of grouping and that type assignment which yields the desired type s.

6. Pronouns

(8) he works, he likes Jane
 $s/(n \setminus s)$ $n \setminus s$ $s/(n \setminus s)$ $n \setminus s/n$ n

(9) that's him, Jane likes him,
 s/n $(s/n) \setminus s$ n $n \setminus s/n$ $(s/n) \setminus s$

 Jane works for him
 n $n \setminus s$ $s \setminus s/n$ $(s/n) \setminus s$

(10) he likes him ;
 $s/(t \backslash s)$ $t \backslash s/t$ $(s/t) \backslash s$

$(s/(t \backslash s))(t \backslash s/t)((s/t) \backslash s)$

(III) $(x/y)(y/z) \rightarrow x/z, \quad (x \backslash y)(y \backslash z) \rightarrow x \backslash z.$

$(s/(t \backslash s))(t \backslash s/t) \rightarrow s/t$

$(t \backslash s/t)((s/t) \backslash s) \rightarrow t \backslash s$

$\underbrace{(\text{he likes})}_{s/\tau_b}$ him, he $\underbrace{(\text{likes him})}_{\tau_b \setminus s}$
 $(s/\tau_b) \setminus s$ $s/(\tau_b \setminus s)$

(John works) for Jane
 τ_b $\tau_b \setminus s$ $s \setminus s/\tau_b$ τ_b

$(\tau_b \setminus s)((s \setminus s/\tau_b)\tau_b) \rightarrow (\tau_b \setminus s)(s \setminus s) \rightarrow \tau_b \setminus s.$

$\tau_b \rightarrow s/(\tau_b \setminus s), \quad \tau_b \rightarrow (s/\tau_b) \setminus s$

$$(IV) \quad x \rightarrow y/(x \setminus y), \quad x \rightarrow (y/x) \setminus y.$$

$$\tau_b^* (\tau_b^* \setminus \tau_b^* / \tau_b') \tau_b \rightarrow (\tau_b^* / \tau_b') \tau_b \rightarrow (\tau_b^* / \tau_b') \tau_b' \rightarrow \tau_b^*$$

From word to sentence: a computational algebraic approach to grammar

Joachim Lambek

<http://www.math.mcgill.ca/barr/lambek/pdffiles/2008lambek.pdf>

I. Introduction

1. Origin of languages.
2. Grammar and mathematics.
3. An algebraic approach to grammar.
4. Mathematical interlude.

basic types

- π = subject,
 π_3 = third person singular subject,
- $\mathbf{s}_1$ = declarative sentence in present tense,
- $\mathbf{i}$ = infinitive of intransitive verb,
- $\mathbf{j}$ = infinitive of any complete verb phrase,
- $\mathbf{o}$ = direct object.

- We postulate
 $\pi_3 \rightarrow \pi, \quad \mathbf{i} \rightarrow \mathbf{j},$

partial order

- *reflexive*: $x \rightarrow x$,
 - *transitive*: $x \rightarrow y$ and $y \rightarrow z$ implies $x \rightarrow z$, and
 - *anti-symmetric*: $x \rightarrow y$ and $y \rightarrow x$ implies $x = y$
-
- We may read $x \rightarrow y$ as saying that everything of type x is also of type y .

$$(xy)z = x(yz)$$

$$1x = x = x1,$$

$$xx^r \rightarrow 1, \quad x^\ell x \rightarrow 1,$$

$$\begin{aligned} \pi_3(\pi^r s_1 \mathbf{j}^\ell)(\mathbf{i} \mathbf{o}^\ell) \mathbf{o} &= [\pi_3 \pi^r] s_1 [\mathbf{j}^\ell \mathbf{i}] [\mathbf{o}^\ell \mathbf{o}] \\ &\rightarrow [\pi \pi^r] s_1 [\mathbf{j}^\ell \mathbf{j}] [\mathbf{o}^\ell \mathbf{o}] \\ &\rightarrow 1 s_1 1 1 = s_1, \end{aligned}$$

she will come

$\pi_3 (\pi^r \mathbf{s}_1 \mathbf{j}') \mathbf{i}$

In the above examples, the pronoun *she* has type π_3 to indicate that it serves as a third person singular subject.

It also has type π , since $\pi_3 \rightarrow \pi$. The intransitive verb *come* has type $\mathbf{i}$ and also $\mathbf{j}$, since $\mathbf{i} \rightarrow \mathbf{j}$.

$\pi_3 (\pi^r \mathbf{s}_1 \mathbf{j}') \mathbf{i} \rightarrow \pi (\pi^r \mathbf{s}_1 \mathbf{j}') \mathbf{i} \rightarrow \pi (\pi^r \mathbf{s}_1 \mathbf{j}') \mathbf{j}$
 $\rightarrow \pi \pi^r \mathbf{s}_1 \mathbf{j}' \mathbf{j} \rightarrow 1 \mathbf{s}_1 1 \rightarrow \mathbf{s}_1$

she will see him

$\pi_3 (\pi^r \mathbf{s}_1 \mathbf{j}') (\mathbf{i} \mathbf{o}') \mathbf{o}$

The transitive verb *see* has type $\mathbf{i} \mathbf{o}'$, to indicate that it is looking for an object on the right.

The superscript l is to show that $\mathbf{o}^l$ cancels $\mathbf{o}$ from the *left*, that is, $\mathbf{o}^l \mathbf{o} \rightarrow 1$.

The finite modal verb-form *will* here has type $\pi^r \mathbf{s}_1 \mathbf{j}'$ to indicate that it is looking for a subject on the left and for an infinitive on the right. The superscript r is to show that π^r cancels π from the *right*, hence also π_3 , since

$$\pi_3 \pi^r \rightarrow \pi \pi^r \rightarrow 1.$$

$$\begin{aligned} \pi_3 (\pi^r \mathbf{s}_1 \mathbf{j}') (\mathbf{i} \mathbf{o}') \mathbf{o} &\rightarrow \pi \pi^r \mathbf{s}_1 \mathbf{j}' \mathbf{i} \mathbf{o}' \mathbf{o} \rightarrow \pi \pi^r \mathbf{s}_1 \mathbf{j}' \mathbf{j} \mathbf{o}' \mathbf{o} \\ &\rightarrow (\pi \pi^r) \mathbf{s}_1 (\mathbf{j}' \mathbf{j}) (\mathbf{o}' \mathbf{o}) \rightarrow 1 \mathbf{s}_1 1 1 \rightarrow \mathbf{s}_1 \end{aligned}$$

she will see him

$\pi_3 (\pi^r \mathbf{s}_1 \mathbf{j}^l) (\mathbf{i} \mathbf{o}^l) \mathbf{o}$

The superscript l again reminds us that $\mathbf{j}^l$ cancels $\mathbf{j}$ from the *left*, hence also $\mathbf{i}$, since

$$\mathbf{j}^l \mathbf{i} \rightarrow \mathbf{j}^l \mathbf{j} \rightarrow 1.$$

$$\begin{aligned} \pi_3 (\pi^r \mathbf{s}_1 \mathbf{j}^l) (\mathbf{i} \mathbf{o}^l) \mathbf{o} &= [\pi_3 \pi^r] \mathbf{s}_1 [\mathbf{j}^l \mathbf{i}] [\mathbf{o}^l \mathbf{o}] \\ &\rightarrow [\pi \pi^r] \mathbf{s}_1 [\mathbf{j}^l \mathbf{j}] [\mathbf{o}^l \mathbf{o}] \\ &\rightarrow 1 \mathbf{s}_1 1 1 = \mathbf{s}_1, \end{aligned}$$

With apologies to Chomsky, I would like to say that the parentheses (and) indicate the surface structure and the square brackets [and] the deep structure.

$$\begin{array}{c}
 \textit{will she see him ?} \\
 (\mathbf{q}_1 \mathbf{j}^\ell \pi^\ell) \pi_3 (\mathbf{i} \mathbf{o}^\ell) \mathbf{o} \rightarrow \mathbf{q}_1 \\
 \underbrace{\hspace{1.5cm}} \quad \underbrace{\hspace{1.5cm}}
 \end{array}$$

Here we have introduced a new basic type
 $\mathbf{q}_1$ = yes or-no question in present tense
 and assigned a new type $\mathbf{q}_1 \mathbf{j}^\ell \pi^\ell$ to *will*.

$$\begin{array}{c}
 \textit{whom will she see - ?} \\
 (\overline{\mathbf{q}} \mathbf{o}^{\ell\ell} \mathbf{q}^\ell) (\mathbf{q}_1 \mathbf{j}^\ell \pi^\ell) \pi_3 (\mathbf{i} \mathbf{o}^\ell) \rightarrow \overline{\mathbf{q}} \\
 \underbrace{\hspace{2.5cm}} \quad \underbrace{\hspace{2.5cm}}
 \end{array}$$

where we have introduced two new basic types

$\mathbf{q}$ = yes-or-no question in any tense,

$\overline{\mathbf{q}}$ = arbitrary question (including wh-question)

and postulated

$$\mathbf{q}_1 \rightarrow \mathbf{q} \rightarrow \overline{\mathbf{q}}.$$

The distinction between q and q is necessary, since a question of type q can be preceded by when of type qq' and a question of type q cannot. The dash at the end of (3.1) was placed where Chomsky would have put a so-called trace.

Traces are not needed in our approach and are only indicated to facilitate comparison with Chomsky's approach. It will turn out that a double superscript I makes an appearance whenever Chomsky requires a trace.

how a sentence such as (3.1) is analyzed
in real time

$$(3.1) \quad \textit{whom will she see} - ?$$

$$(\bar{q}o^{\ell\ell}q^{\ell})(q_1j^{\ell}\pi^{\ell})\pi_3(io^{\ell}) \rightarrow \bar{q}$$

$$\begin{aligned} \textit{whom} &: \bar{q}\hat{o}^{\ell\ell}q^{\ell}, \\ \textit{whom will} &: \bar{q}\hat{o}^{\ell\ell}\underline{q^{\ell}q_1}j^{\ell}\pi^{\ell} \rightarrow \bar{q}\hat{o}^{\ell\ell}j^{\ell}\pi^{\ell}, \\ \textit{whom will she} &: \bar{q}\hat{o}^{\ell\ell}j^{\ell}\pi^{\ell}\pi_3 \rightarrow \bar{q}\hat{o}^{\ell\ell}j^{\ell}, \\ \textit{whom will she see} &: \bar{q}\hat{o}^{\ell\ell}j^{\ell}\underline{i o^{\ell}} \rightarrow \bar{q}\hat{o}^{\ell\ell}\underline{o^{\ell}} \rightarrow \bar{q}. \end{aligned}$$

理論とモデル

意味の二項性

二つの言語で意味を考える

- 「私はあなたを愛しています。」 この文の「意味」は、何かと言われると、なかなか答えるのが難しい。ただ、"I love you." の「意味」はと聞かれれば、「私はあなたを愛しています。」だと答えるのは易しい。
- 「それは、意味の意味が違う。」 確かに、そうかもしれない。それでは、「二つの言語で、同じ意味が表現されている」と考えるのは、どうなのだろうか？ この文章で使われている「意味」は、先に「なかなか答えるのが難しい」と考えた「意味」そのものではないだろうか？
- とりあえず、二つのことを、この後の議論のために、作業仮説として確認しておこう。

二つの言語を比較すると、意味は取り出しやすく(感じる)。
意味は言語によって表現されるが言語によらないものを指し示す。

語の意味と文の意味

- 言葉の意味を調べようとするとき、我々に一番身近な行為は、「辞書」を引くことである。ただし、辞書が与えるのは、語の意味である。
- 翻訳が文を対象にするのに対して、辞書は語を対象にする。文は語からできているので、辞書が与える語の意味の情報は、その語を含む文の意味を考えるもっとも基本的な情報を与える。
- 辞書で、I = 私、love = 愛する、you = あなた という情報が得られたとしよう。この情報だけから、"I love you"という文を翻訳すると、「私 愛する あなた」になるのだが、これはどうも日本語としてはうまくない。
- その理由は、はっきりしている。

言語の意味についての基本的な問題

- 語から文が構成されているのは明らかなのだが、語から文を構成するときに、日本語でも英語でも、ある構成規則に従う。それを「文法」という。文法は、ある言語での語の出現順序に強い制限を与える。「私 愛する あなた」は、日本語の語の出現順序にそぐわないのだ。
- 先に、簡単に「文は語からできている」といった。それはそれで間違いではない。もう少し正確に言えば、「文は、文法という構成規則に従って、語から構成される」ということになる。
- 「異なる語から異なる構成規則で構成された文が「同じ意味」を持ちうるのはなぜか？」というのは、言語の意味についての最も重要な問題なのである。

意味の「二項性」

- ある二つのものの関係の中で、あるものを考えるのは、あるものだけをじっと見ているより、普段は気づくことのないものへの気づきが生まれると僕は考えている。
- 意味についてのアプローチは、様々あるのだけれど、ソシュールのシニフィアン(signifiant: 意味するもの)とシニフィエ(signifié: 意味されるもの)にしても、オグデン＝リチャードの「意味の三角形」の「シンボル」と「指示するもの」「指示されるもの」にしても、こうした「二項関係」が基本になっている。(三角なのに二項なのかというツツコミは、あとでこたえる)
- こうした二項関係は、もっと深いところでは、「主体」と「客体」、「認識するもの」と「認識されるもの」という認識の構造そのものに基礎を持っている。

「ルール」の世界と「たとえ」の世界

「ルール」で理解する vs. 「たとえ」で理解する

2+3が5になるのを、「りんご二つと、みかん三つで、全部で五つ」というように、具体的なものをあげて、「たとえ」で説明することができる。

数学のルールは、基本的には現実の具体的な関係から抽象されたものであるので、理解しやすい具体的関係から、ルールを導入・説明するのは有効である。

「ルール」は、抽象的なものだが、「たとえ」は、ルールの「意味」を理解させる。

「ルール」で理解する vs. 「たとえ」で理解する

先のりんごとみかんの例では、実は、「全部で」という言葉が、足し算の説明になっている。「たとえ」では、具体的なものにたいする理解とともに、「言葉」の理解が、前提になることがある。

「たとえ」の世界は、具体的で現実的である。その世界は、我々の直観と「言葉」によって理解可能な「意味」の世界によって支えられている。

ただ、それは、複数の意味を持つかもしれないし、ひとつの「たとえ」で全ての「ルール」を説明できないという意味では個別的である。

また「たとえ」にはいろいろのタイプがある。言葉だけでなく、図解すること、また「たとえ」による説明の一種である。言葉による「たとえ」と図形による「たとえ」では、抽象度が異なる。

「ルール」の世界と「たとえ」の世界

数学の計算では「ルール」を「アルゴリズム」という。その「ルール」に従って、きちんと計算すると、余計なことは考えなくても正しい計算結果が保証されるという限りでは、「ルール」の世界は、他の世界とは独立しているように見える。

ただ、「ルール」は、計算をするという目的のためには万能かもしれない。ただ、計算「ルール」というものは、具体的なようで我々の頭の中にしかない抽象的なものであって、それ自体では意味を持たないようにも見える。

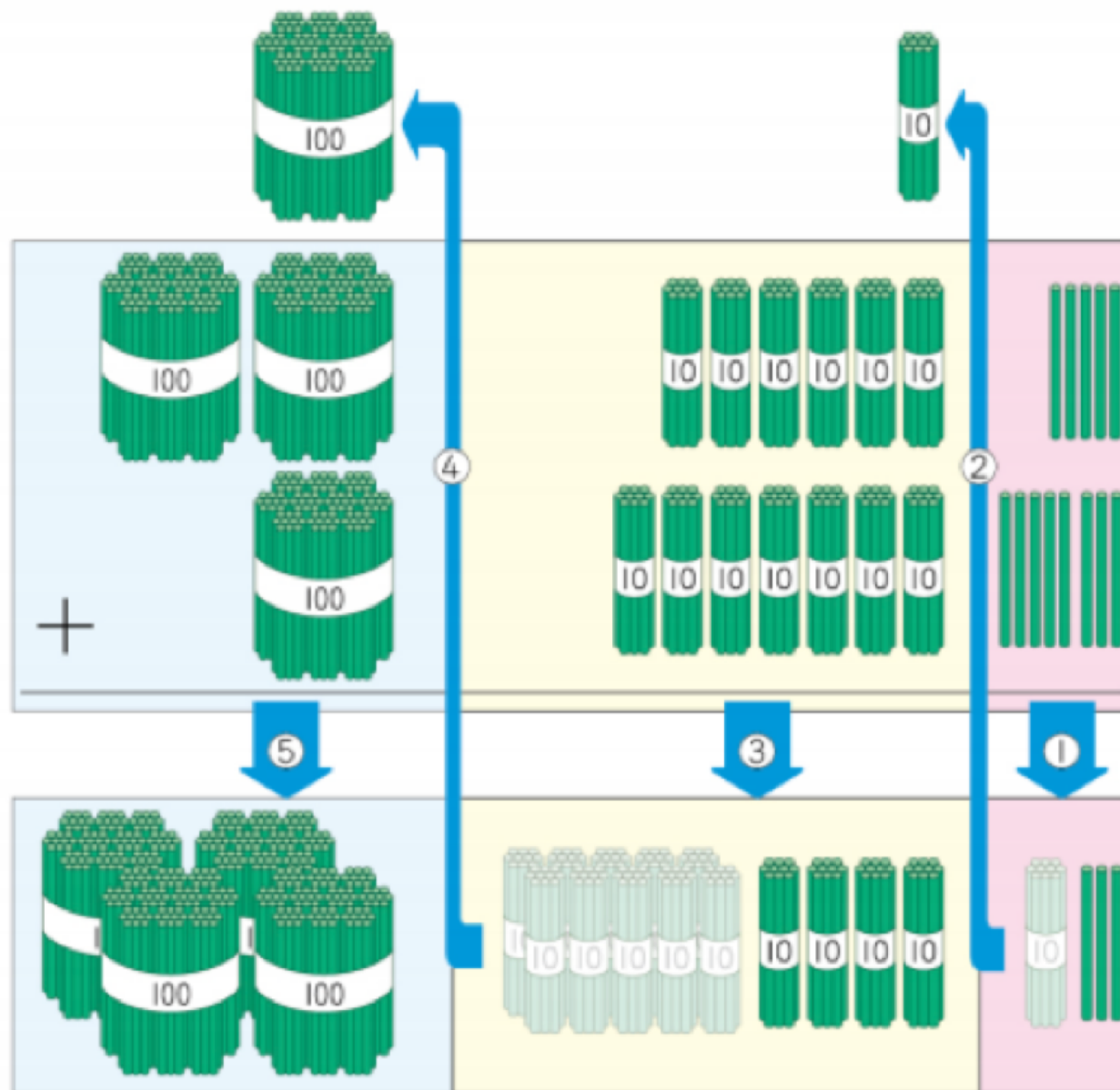
数学は、基本的には、それ自身の原理に基づいた自律的な「ルール」の世界である。(実際には、それは単純化しすぎである。)

3

265+178 の筆算のしかたを考えましょう。

啓林館

3年上 p.57



$$\begin{array}{r} 265 \\ + 178 \\ \hline 3 \end{array}$$

$$\begin{array}{r} 265 \\ + 178 \\ \hline 43 \end{array}$$

$$\begin{array}{r} 265 \\ + 178 \\ \hline 443 \end{array}$$

$$\frac{1}{2} \div \frac{2}{3} = \frac{3}{4} \text{ の場合}$$

鉛筆が12本入ったケースと、3人で1組のグループがあったとする。この時、ケースの1/2をグループの2/3で分けたとしよう。

これは、鉛筆6本を二人で分けたことになるので、グループの一人は3本(3/12)の鉛筆を取ったことになる。この取り方で、一つのグループあたり何本の鉛筆が分配されたかを考えれば、グループは、3人で1組なので、9本の鉛筆をえたことになる。これは、ケースで考えれば、 $9/12=3/4$ ケースになる。

1グループあたりのケースを求めると、

$$\frac{1}{2} \div \frac{2}{3} = \frac{1}{2 \times 2} \times 3 = \frac{1 \times 3}{2 \times 2} = \frac{3}{4}$$

4

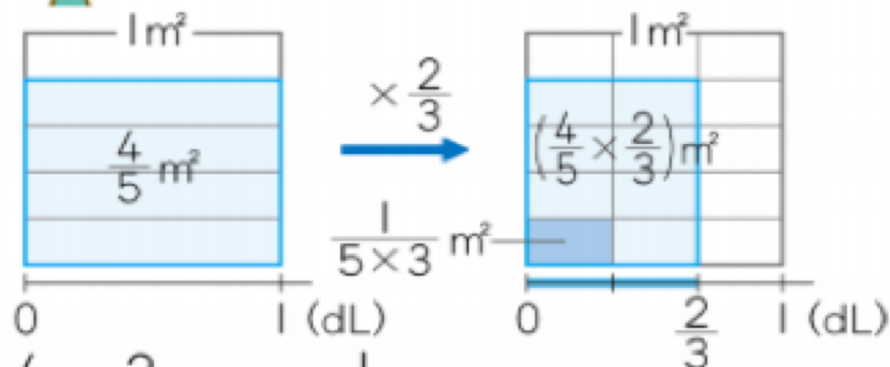
1 dLで $\frac{4}{5}\text{m}^2$ ぬれるペンキがあります。
 $\frac{2}{3}\text{dL}$ のペンキでは何 m^2 ぬれますか。



6年 p.280



ひなた



$\frac{4}{5} \times \frac{2}{3}$ は、 $\frac{1}{5 \times 3}$ の が

(4×2)個分だから、

$$\frac{4}{5} \times \frac{2}{3} = \frac{\boxed{} \times \boxed{}}{\boxed{} \times \boxed{}}$$

$$= \boxed{} \text{ m}^2$$



だいち

$$\begin{aligned} \frac{4}{5} \times \frac{2}{3} &= \frac{4}{5} \times \left(\frac{1}{3} \times 2 \right) \\ &= \left(\frac{4}{5} \times \frac{1}{3} \right) \times 2 \\ &= \frac{4}{5 \times 3} \times 2 \end{aligned}$$

$$= \frac{\boxed{} \times \boxed{}}{\boxed{} \times \boxed{}}$$

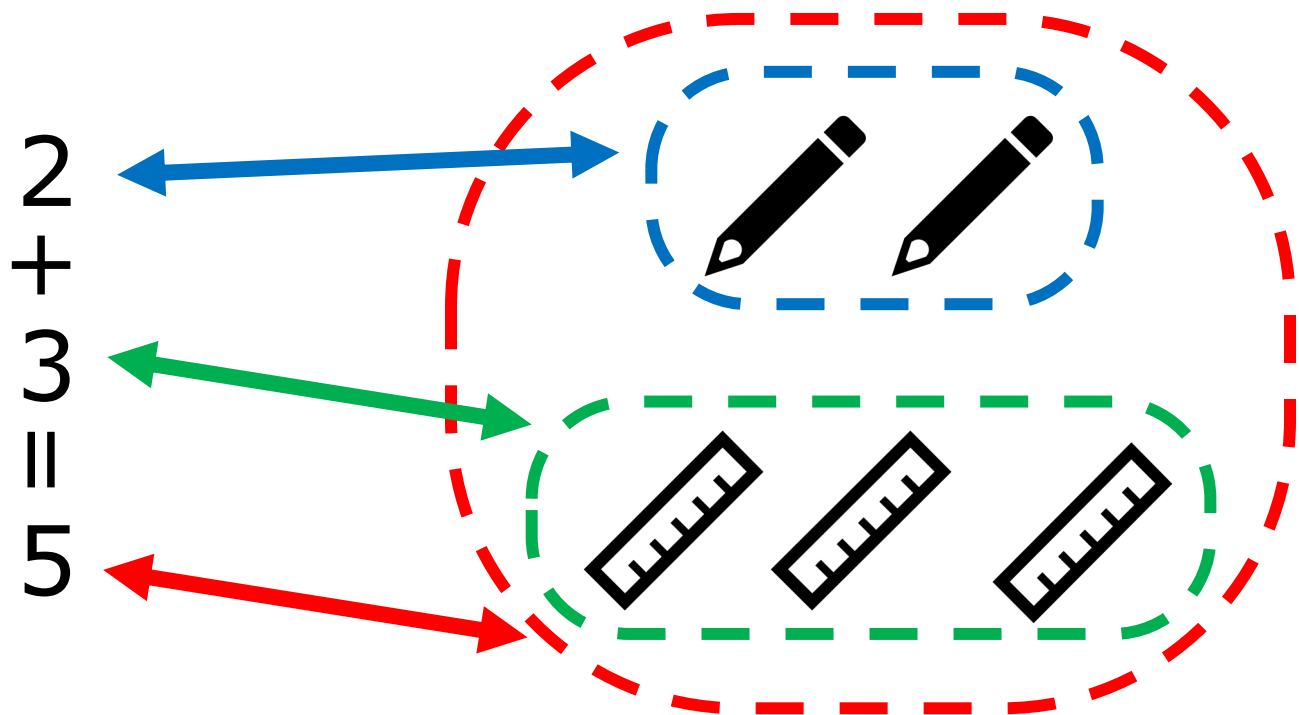
$$= \boxed{} \text{ m}^2$$

語の意味のモデル

素朴な対応理論

「ルール」と「たとえ」の対応関係

先に見た「ルールの世界」と「たとえの世界」の二つの世界は、お互いに関連している。「ルールの世界」は「たとえの世界」に、具体的対応物を持ち、「たとえの世界」は「ルールの世界」に抽象的な対応物を持つ。

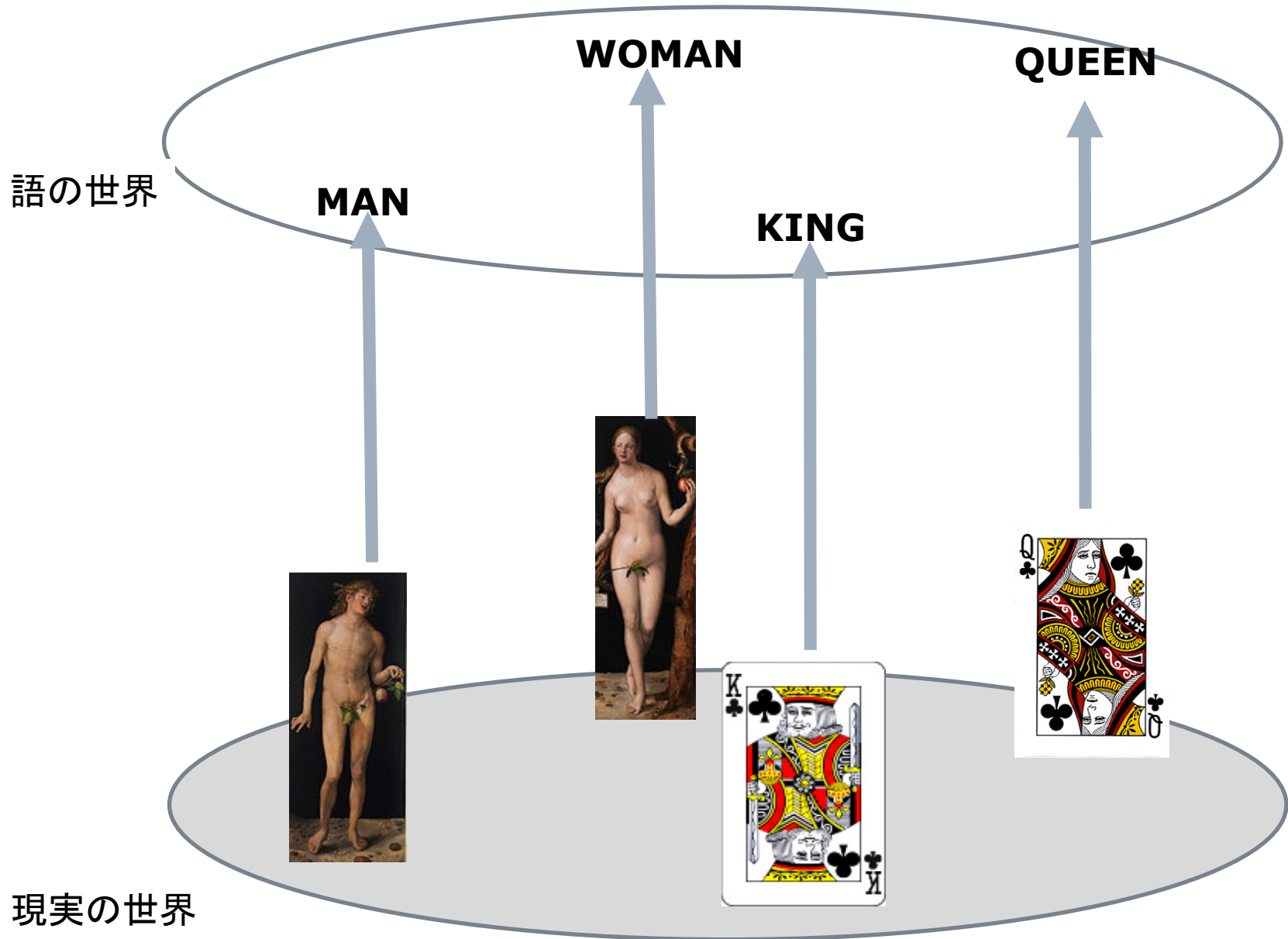


二つの「世界」の対応関係が、「意味」を与える

二つの世界に対応関係があり、一方の世界がより具体的で、もう一方の世界がより抽象的な場合、この対応関係は、抽象的な世界に、具体的な「意味」を与える。あるいは、この対応関係は、抽象的なものに具体的な「解釈」を与える。

以下では、「語の意味」の素朴なモデルとして、現実に存在する「事物」と、その「名前」の「対応関係」を、考えてみよう。

「語の意味」の素朴なモデル



意味：ものに「名前」をつけること

語の世界

MAN

WOMAN

QUEEN

KING

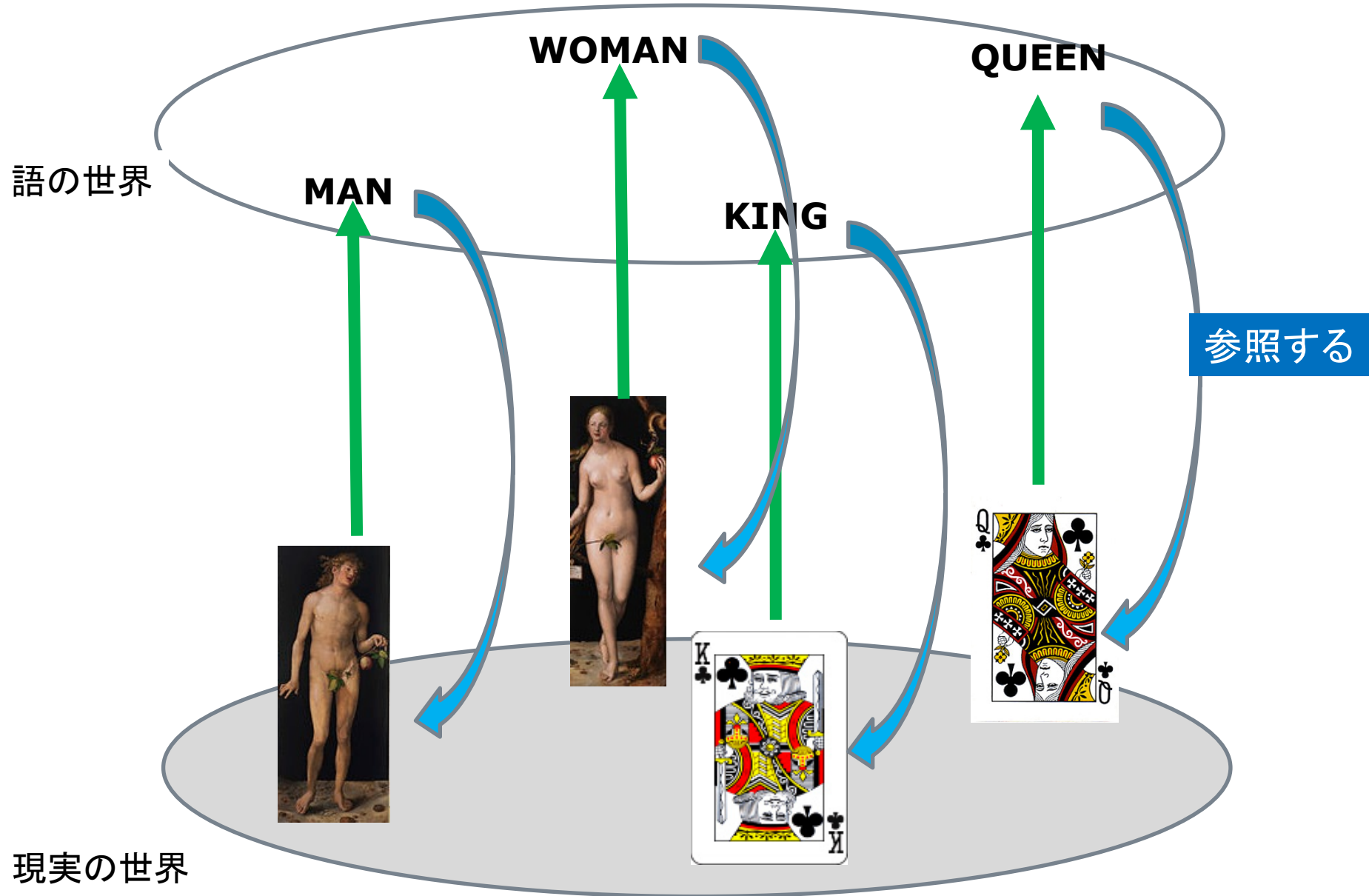
対応づける
名付ける

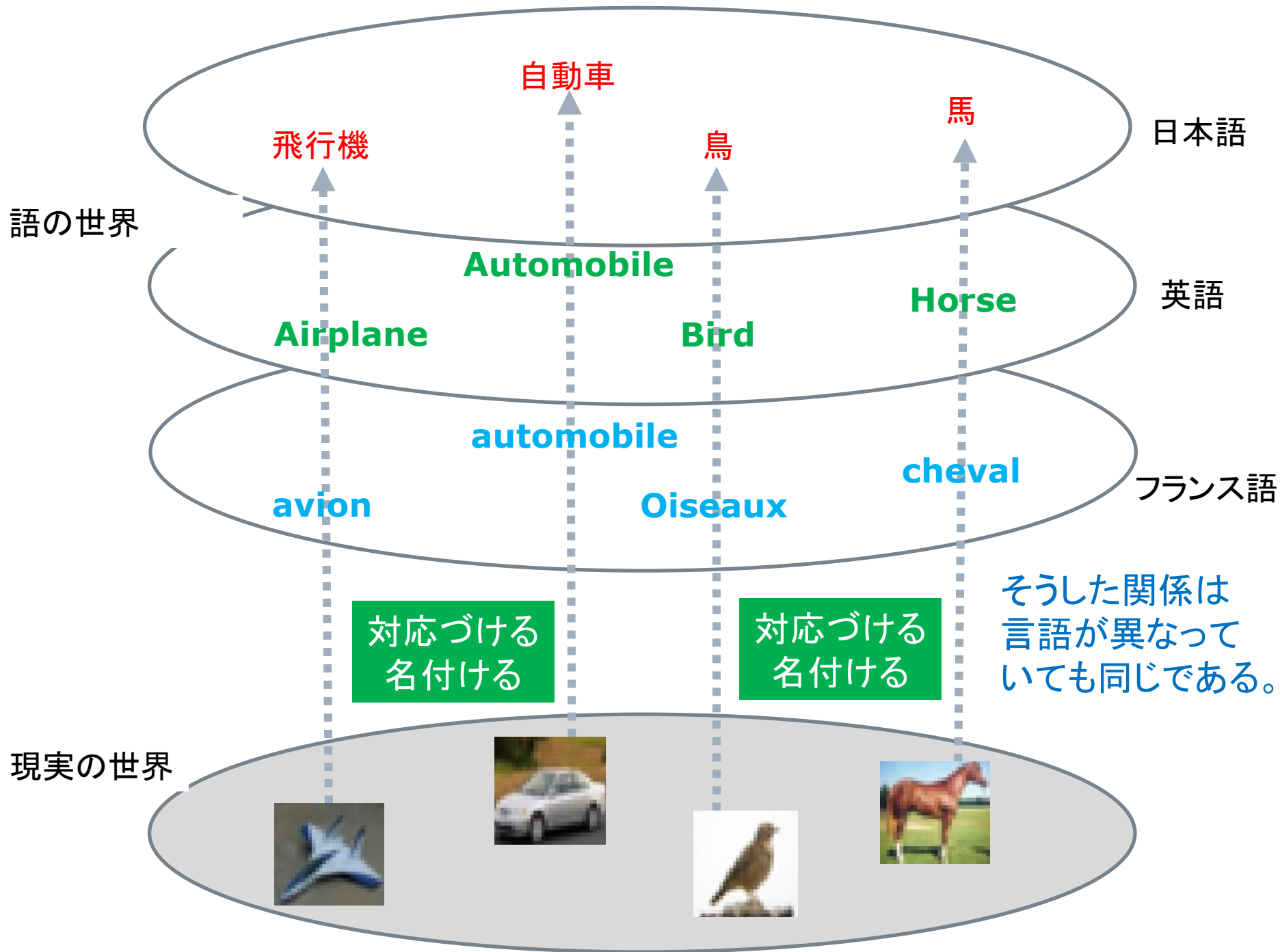
対応づける
名付ける

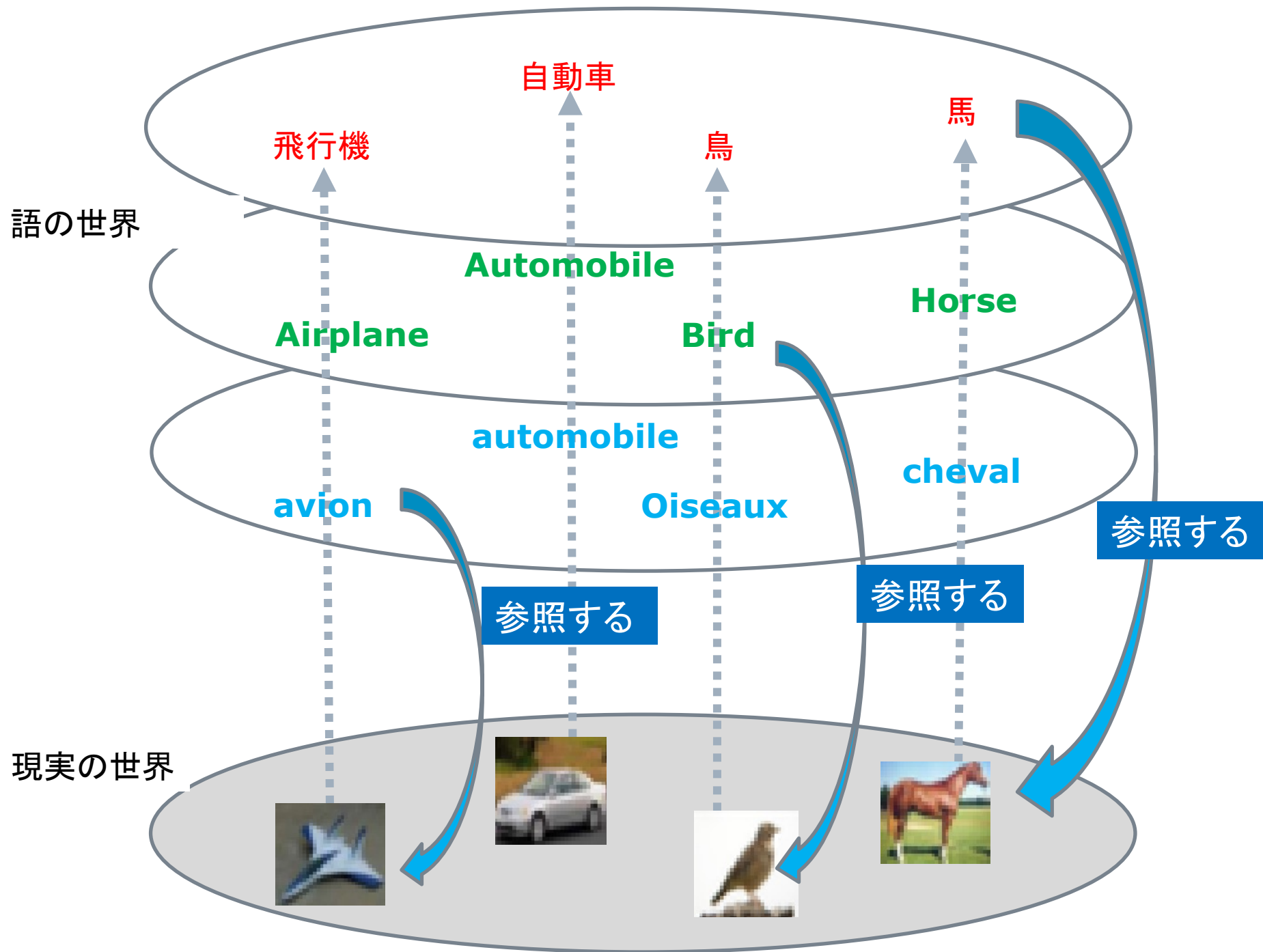


現実の世界

語は、ものを「参照」する







素朴な「対応理論」の弱点

- ただ、こうした素朴な「対応理論」には、弱点もある。
「現実の世界での『馬』」という対象を考えてみよう。こうした対象の「実在性」については、は、いろいろな疑問がうまれる。「現実中存在するのは、個々の具体的なウマだけである。」「いや、『馬』の意味の実在性には、疑問の余地はない。」等々。
- その上、語は(文になればなおさらなのだが)、「物」に付けられた意味を持つ名前だけではない。抽象的な概念(「自由」「平等」「意味」...)や、物ではない「出来事」(「第二次世界大戦」「爆発」...)も名前を持つのだ。具体的な心理的な状態(「悲しみ」「意図」...)も。これらを先の図にマッピングするのは、難しい。
- 「状態の変化」の一群は、「動詞」で表現される。

現実の世界

「同一性」

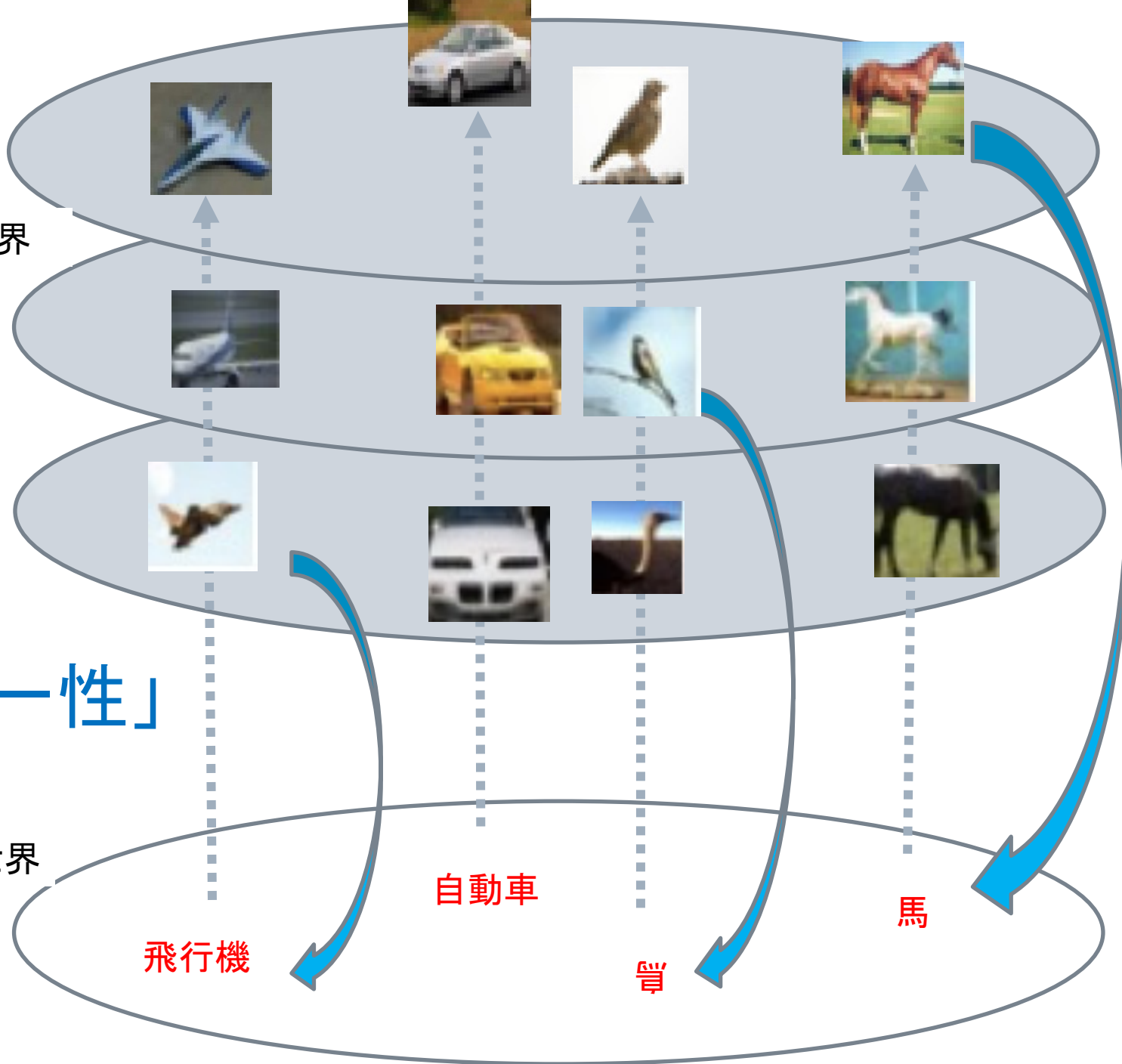
意味の世界

飛行機

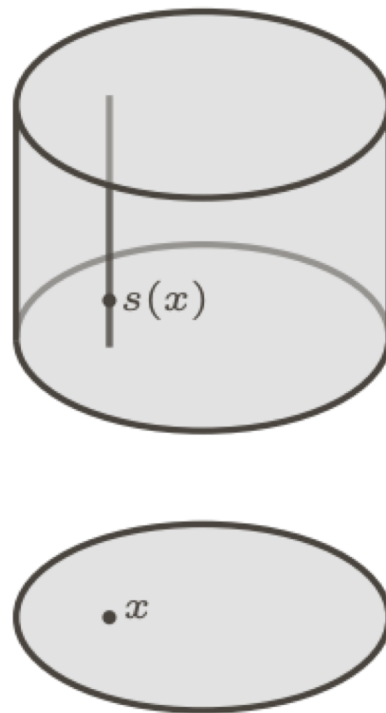
自動車

翼

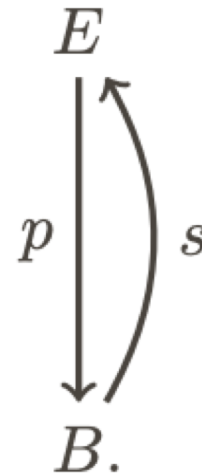
馬



新しい型の理論: 関数型言語のdependent type
は、homotopy theoryに対応物を持つ



Fibration / Section



<i>type theory</i>	<i>homotopy theory</i>
$(x : B) \quad E(x)$	$p : E \rightarrow B$ is a fibration over B
$(x : B) \quad s(x) : E(x)$	s is a section of p

Homotopy Type Theory and Voevodsky's Foundation

Recent discoveries have been made connecting abstract homotopy theory and the field of type theory from logic and theoretical computer science. This has given rise to a new field, which has been christened “homotopy type theory”. In this direction, Vladimir Voevodsky observed that it is possible to model type theory using simplicial sets and that this model satisfies an additional property, called the Univalence Axiom, which has a number of striking consequences. He has subsequently advocated a program, which he calls univalent foundations, of developing mathematics in the setting of type theory with the Univalence Axiom and possibly other additional axioms motivated by the simplicial set model. Because type theory possesses good computational properties, this program can be carried out in a computer proof assistant.

<https://arxiv.org/pdf/1210.5658.pdf>

理論とモデル

対応の形式化

「理論」と「モデル」の形式的定義

ここでは、「ルールの世界」と「たとえの世界」の数学的形式化を紹介したい。

ある世界で、基本的な公理と一階の述語論理の演繹規則が与えられているとしよう。この世界を「理論 Theory」と呼ぶことにしよう。この「理論」の世界で、ある命題 ϕ が成り立つのは、この理論の中で、命題 ϕ が証明される場合に限る。

今度は、もう一つ別の世界を考えよう。この世界では、ある命題 ϕ が妥当する（「真」である）ことが、何らかの方法で定義されているとする。こうした世界を「モデル Model」と呼ぼう。

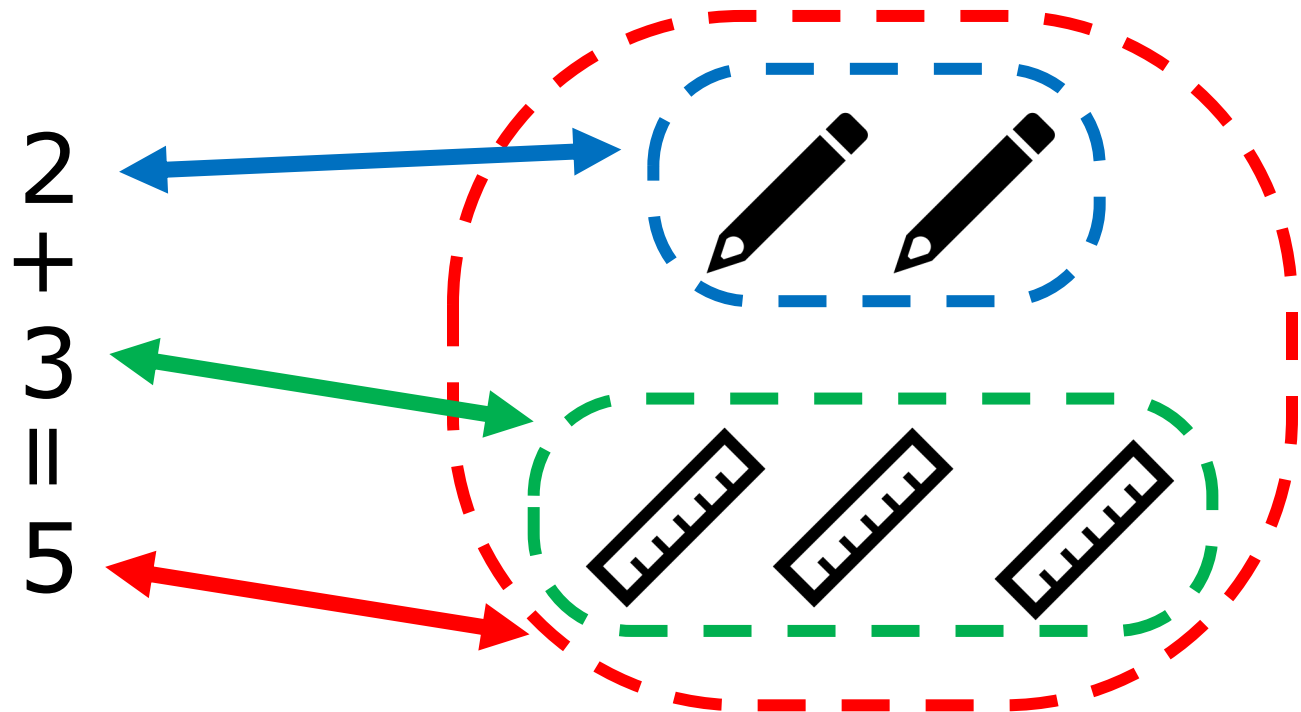
ある理論 T とあるモデル M について、すべての命題 ϕ について、次の関係が成り立つとする。

T において ϕ は証明可能 $\rightarrow M$ において ϕ は真である

この時、 T はモデル M を持つ。あるいは、 M は T のモデルである。という。

「ルール」の世界と「たとえ」の世界

「ルールの世界」と「たとえの世界」の二つの世界は、お互いに関連している。「ルールの世界」は「たとえの世界」に、具体的対応物を持ち、「たとえの世界」は「ルールの世界」に抽象的な対応物を持つ。



「理論」の世界と「モデル」の世界

理論 T

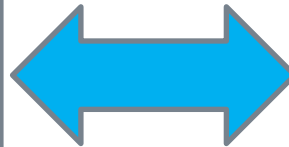
形式的なルールが
与えられている。
その下で

命題 ϕ は証明可能

モデル M

なんらかのルールが
この世界では成り立っていて
その下で

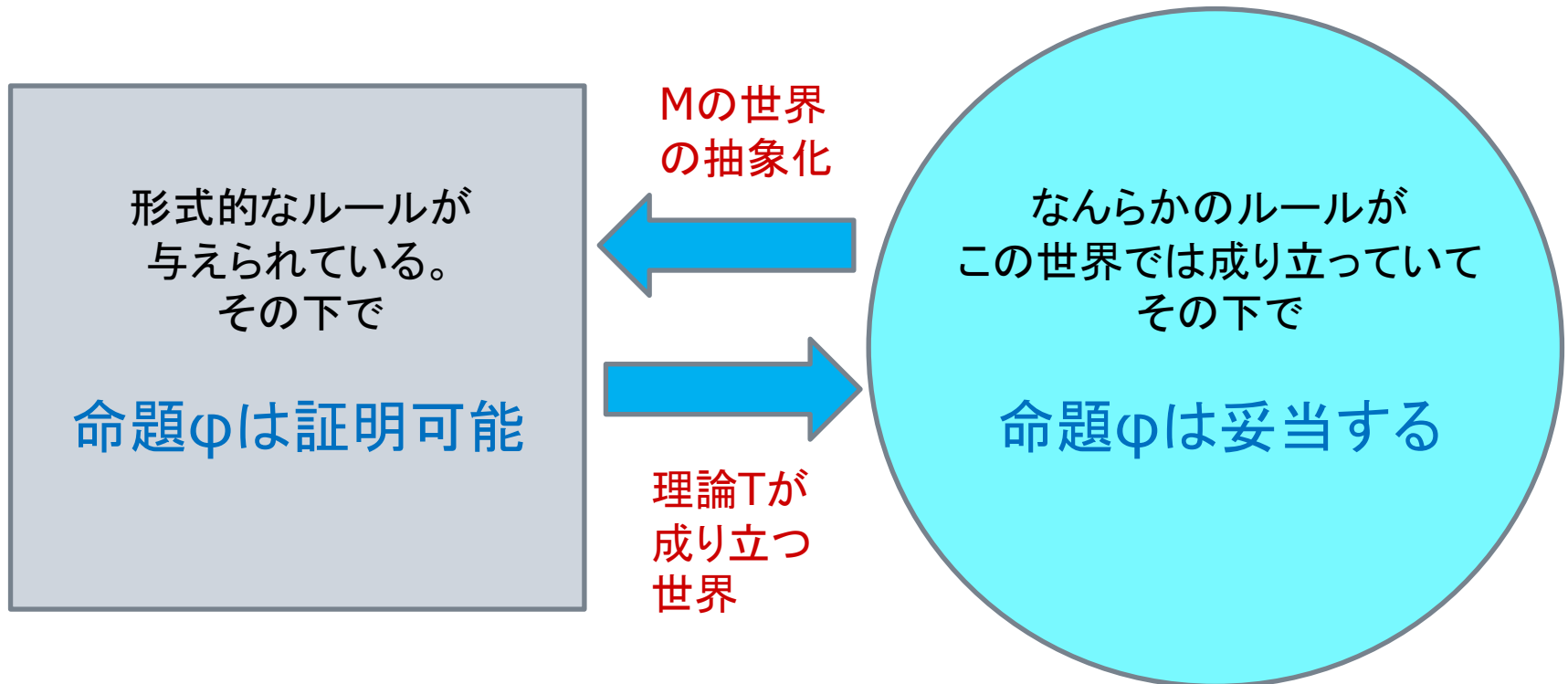
命題 ϕ は妥当する



「理論」の世界と「モデル」の世界

理論 T

モデル M

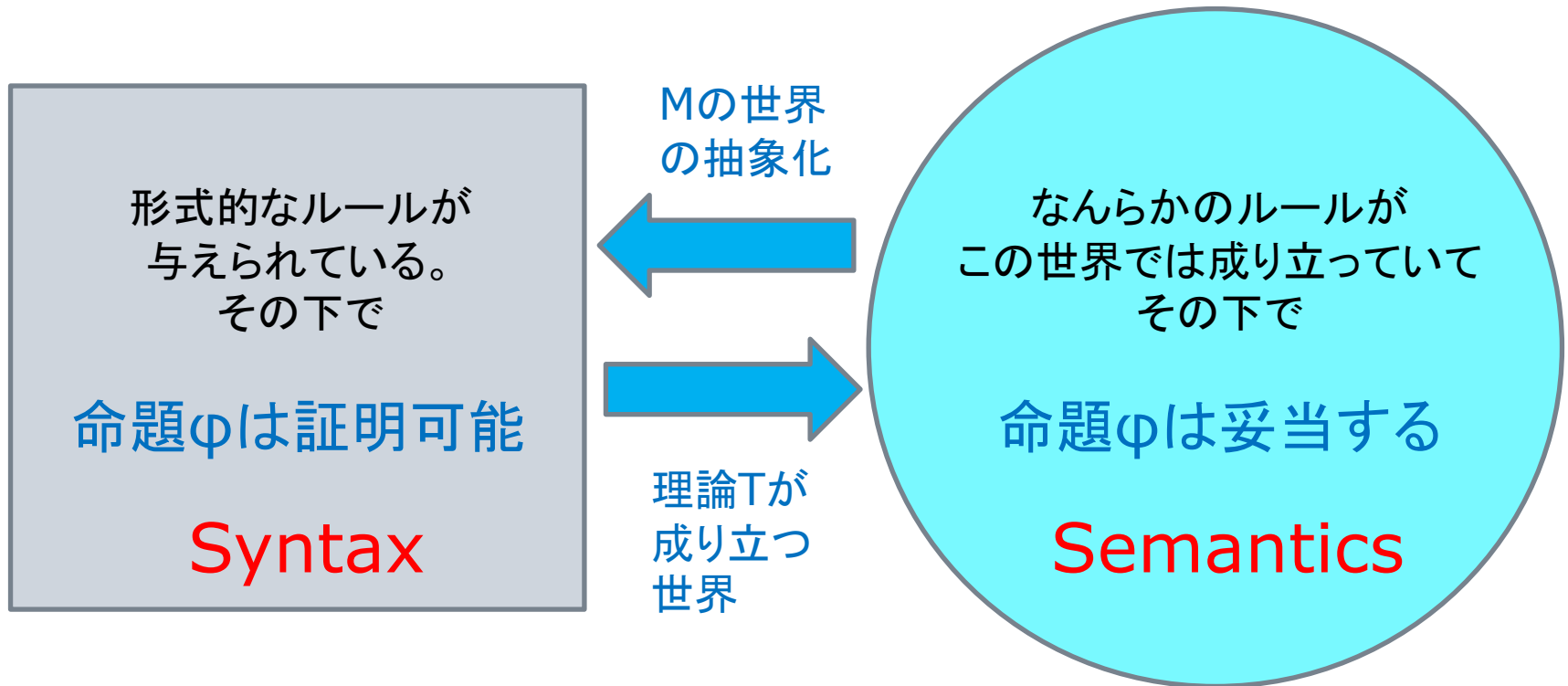


理論とモデルの「二重化」あるいは「双対性」

「理論」の世界と「モデル」の世界

理論 T

モデル M



モデルMは、理論Tに「意味」を与える

ゲーデルの完全性定理

理論とモデルの関係を、「すべてのモデルで真になる命題は、一階の述語論理で証明可能である」という形で初めて述べたのは、ゲーデルである。これをゲーデルの「完全性定理」という。

この定理から、「ある理論が無矛盾ならそれはモデルを持つ」ことが導かれる。<https://goo.gl/5HLMsK>

モデルの存在定理では、次の「レーベンハイム＝スコーレムの定理」が有名である。

「理論が非可算濃度のモデルを持つなら、それは可算モデルも持つ。」

レーベンハイム・スコレムの定理

「モデルをもつなら、可算モデルも持つ」

Skolem's paradox is that

“every countable axiomatisation of set theory in first-order logic, if it is consistent, has a model that is countable.”

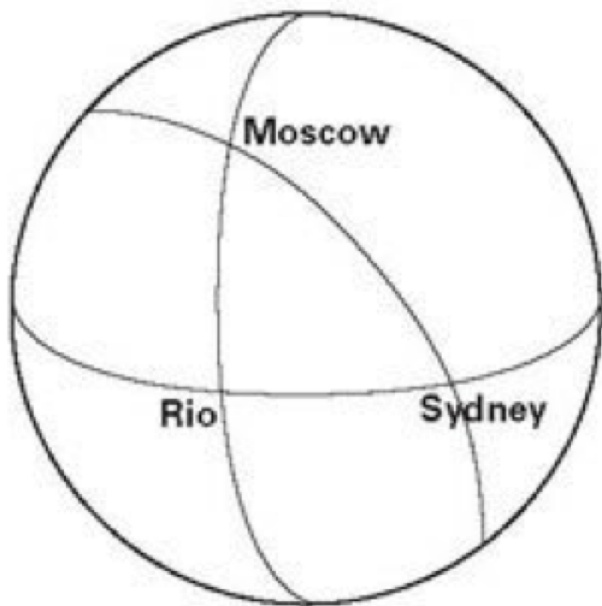
This appears contradictory because it is possible to prove, from those same axioms, a sentence that intuitively says (or that precisely says in the standard model of the theory) that there exist sets that are not countable. Thus the seeming contradiction is that a model that is itself countable, and which therefore contains only countable sets, satisfies the first order sentence that intuitively states "there are uncountable sets".

理論とモデル 2

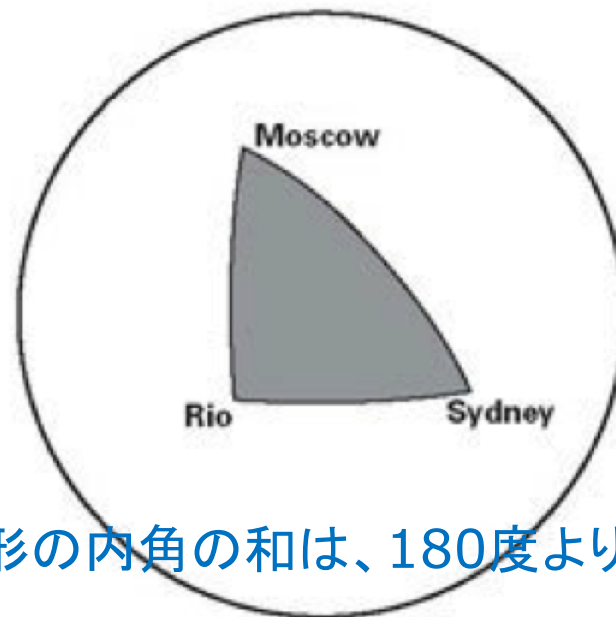
数学でのモデルの利用

非ユークリッド幾何学の発見

- Lobachevskyは平行な仮定を否定することによって非ユークリッド幾何学を作成したが、Bolyaiはユークリッド幾何学と双曲幾何学の両方が、パラメータ k に依存するような幾何学を作り出した。Bolyaiは物理宇宙の幾何学的構造がユークリッド的または非ユークリッド的であるとしても、それを数学的推論だけで決めることは不可能であるとして、それは物理学の課題だと述べて、彼の研究を終える。



楕円幾何学

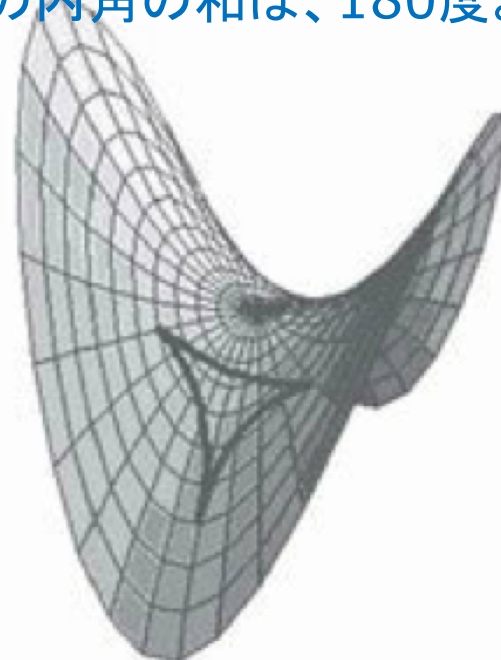


三角形の内角の和は、180度より大きい

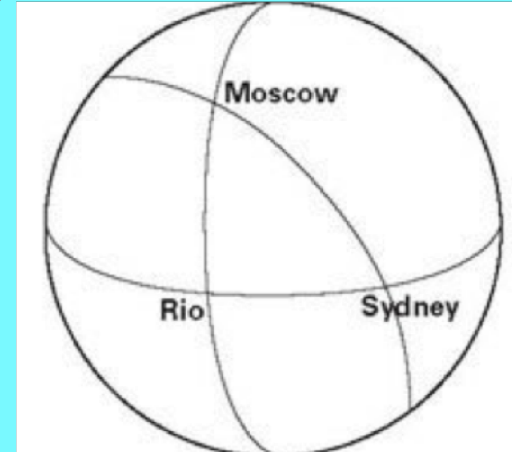
三角形の内角の和は、180度より小さい

双曲幾何学

リーマン幾何学



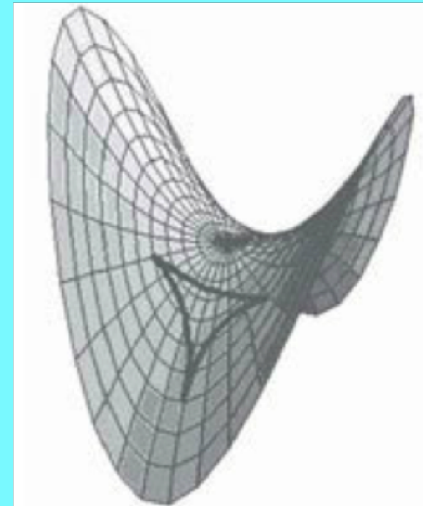
三角形の内角の
和が、180度より
大きい幾何学



理論

モデル

三角形の内角の
和が、180度より
小さい幾何学



THE CONSISTENCY OF THE CONTINUUM HYPOTHESIS

KURT GÖDEL

ANNALS OF MATHEMATICS STUDIES
PRINCETON UNIVERSITY PRESS

²⁰ Gardner, R. S., A. J. Wahlen, C. Basilio, R. S. Miller, P. Lengyel, and J. F. Speyer, these PROCEEDINGS, 48, 2087 (1962).

²¹ Ralph, R. K., and H. G. Khorana, *J. Am. Chem. Soc.*, **83**, 2926 (1961).

²² Ames, B. N., and D. T. Dubin, *J. Biol. Chem.*, **235**, 769 (1960).

²³ Chen, P. S., Jr., T. Y. Toribara, and H. Warner, *Anal. Chem.*, **28**, 1756 (1956).

²⁴ Heppel, L. A., D. R. Harkness, and R. J. Hilmeo, *J. Biol. Chem.*, **237**, 841 (1962).

²⁵ Russell, W. E., and H. G. Khorana, *J. Biol. Chem.*, **234**, 2114 (1959).

²⁶ Kuy, E. R. M., N. S. Simmons, and A. L. Doucet, *J. Am. Chem. Soc.*, **74**, 1724 (1952).

²⁷ Sherman, J. R., and J. Adler, *J. Biol. Chem.*, **238**, 873 (1962).

²⁸ Waley, S. G., and J. Watson, *Biochem. J.*, **55**, 328 (1953).

²⁹ Akabori, S., K. Ohno, and K. Narita, *Bull. Chem. Soc. Japan*, **25**, 214 (1952).

³⁰ Lowry, O. H., N. J. Rosebrough, A. L. Farr, and R. J. Randall, *J. Biol. Chem.*, **193**, 265 (1951).

³¹ Nakamoto, T., and S. B. Weiss, these PROCEEDINGS, 48, 880 (1962).

³² Fox, C. F., W. S. Robinson, and S. B. Weiss, *Fed. Proc.*, **22**, 463 (1963).

³³ Krakow, J. S., and S. Oshon, these PROCEEDINGS, 49, 88 (1963).

THE INDEPENDENCE OF THE CONTINUUM HYPOTHESIS

By PAUL J. COHEN*

DEPARTMENT OF MATHEMATICS, STANFORD UNIVERSITY

Communicated by Kurt Gödel, September 30, 1963

This is the first of two notes in which we outline a proof of the fact that the Continuum Hypothesis cannot be derived from the other axioms of set theory, including the Axiom of Choice. Since Gödel¹ has shown that the Continuum Hypothesis is consistent with these axioms, the independence of the hypothesis is thus established. We shall work with the usual axioms for Zermelo-Fraenkel set theory,² and by Z-F we shall denote these axioms without the Axiom of Choice, (but with the Axiom of Regularity). By a model for Z-F we shall always mean a collection of actual sets with the usual ϵ -relation satisfying Z-F. We use the standard definition³ for the sets of integers ω , ordinals, and cardinal numbers.

The purpose of this note is to show that the Continuum Hypothesis is not derivable from the axioms of Z-F. The second note will show that the Continuum Hypothesis is not derivable from the axioms of Z-F plus the Axiom of Constructibility. In the course of reference 3, let the Axiom of Choice and the Generalized Continuum Hypothesis both hold.

(2) The continuum (i.e., $\mathfrak{P}(\omega)$ where $\mathfrak{P}$ means power set) has no well-ordering.

(3) The Axiom of Choice holds, but $\aleph_1 \neq 2^{\aleph_0}$.

(4) The Axiom of Choice for countable pairs of elements in $\mathfrak{P}(\mathfrak{P}(\omega))$ fails.

Only part 3 will be discussed in this paper. In parts 1 and 3 the universe is well-ordered by a single definable relation. Note that 4 implies that there is no simple ordering of $\mathfrak{P}(\mathfrak{P}(\omega))$. Since the Axiom of Constructibility implies the Generalized Continuum Hypothesis,⁴ and the latter implies the Axiom of Choice,⁵ Theorem 1 completely settles the question of the relative strength of these axioms.

Before giving details, we sketch the intuitive ideas involved. The starting point is the realization^{1, 4} that no formula $\phi(x)$ can be shown from the axioms of Z-F to have the property that the collection of all x satisfying it form a model for Z-F in which the Axiom of Constructibility ($V = L$,⁶) fails. Thus, to find such models, it seems natural to strengthen Z-F by postulating the existence of a set which is a

非カントールの集合論の発見

ゲーデル

連続体仮説と集合論の無矛盾性の証明

- 「世紀の難問」と呼ばれた連続体仮説の問題で、大きな前進があったのは、1940年のことだ。クルト・ゲーデルが、連続体仮説と選択公理の二つを満たす集合論のモデルを実際に構成して見せたのだ。この集合論のモデルの中では、連続体仮説は「真」である。
 - この結果は、連続体仮説を直接に証明したものではないことに注意しよう。それは、集合論と連続体仮説には矛盾がないこと、すなわち、集合論の中では連続体仮説の否定は証明できないことを意味している。
-

コーエン 1963年

連続体仮説と集合論の独立性の証明

- 連続体仮説を最終的に解決したのは、J.P.コーエンだった。彼は、「強制法(Forcing Method)」というモデル構成法を創出し、連続体仮説の否定が成り立つモデルを構成した。
 - 先のゲーデルの結果と合わせると、連続体仮説は、集合論の公理群からは独立であることが証明されたことになる。連続体仮説は、集合論の公理群からは証明できないのだ。
 - 結果から見ると、カントールが、全力で連続体仮説を証明しようとして、成功しなかったのには理由があったのである。
-

ゲーデル
連続体仮説(GCH)
と集合論(ZF)の
無矛盾性証明



連続体仮説が
成り立つ集合論の
モデルの構築

コーエン
連続体仮説(GCH)
の集合論(ZF)からの
独立性証明



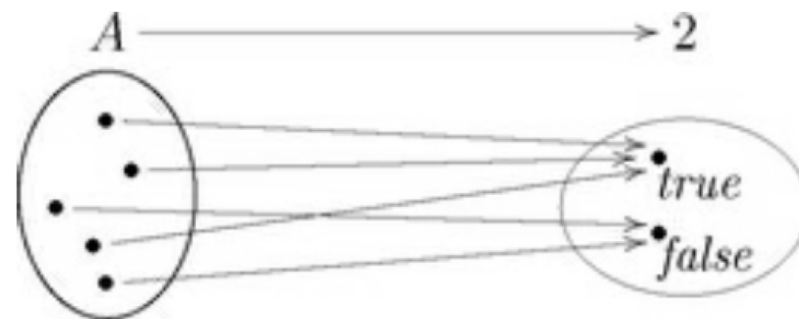
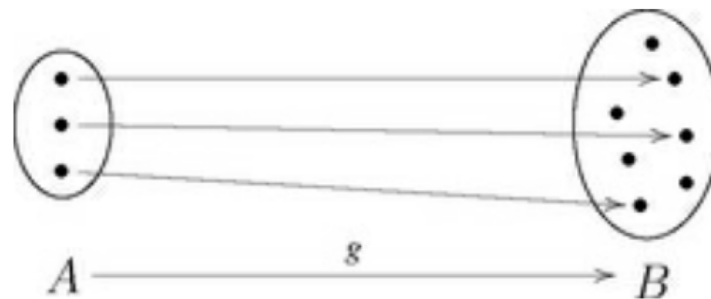
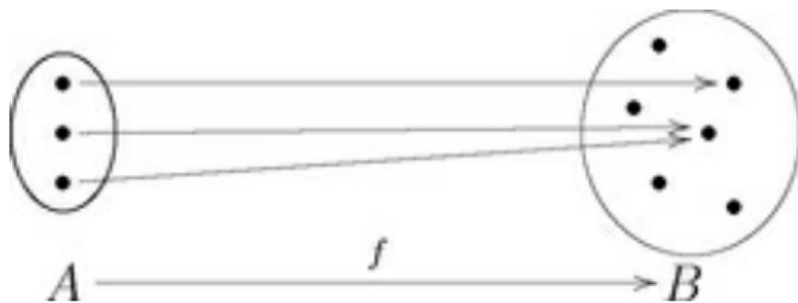
連続体仮説の否定が
成り立つ集合論の
モデルの構築

Category論と Functor Semantics

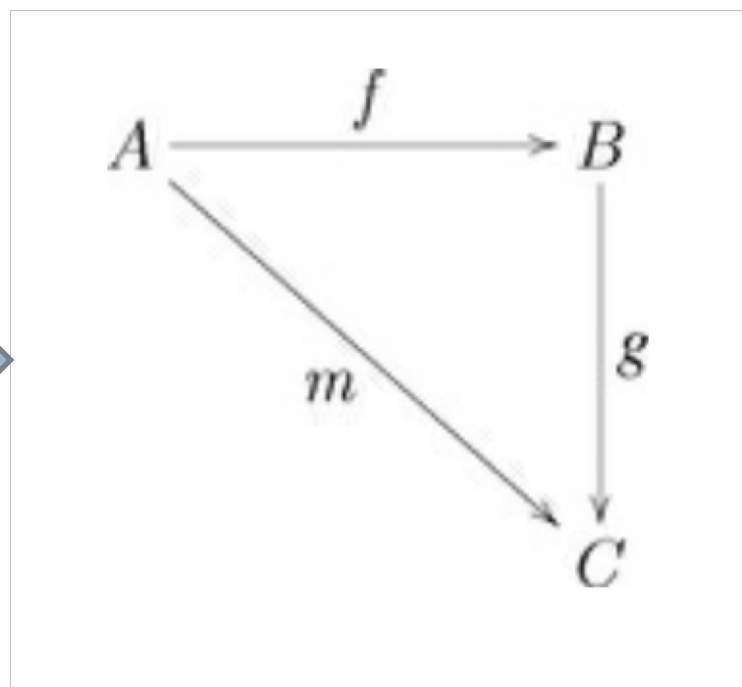
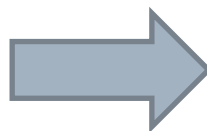
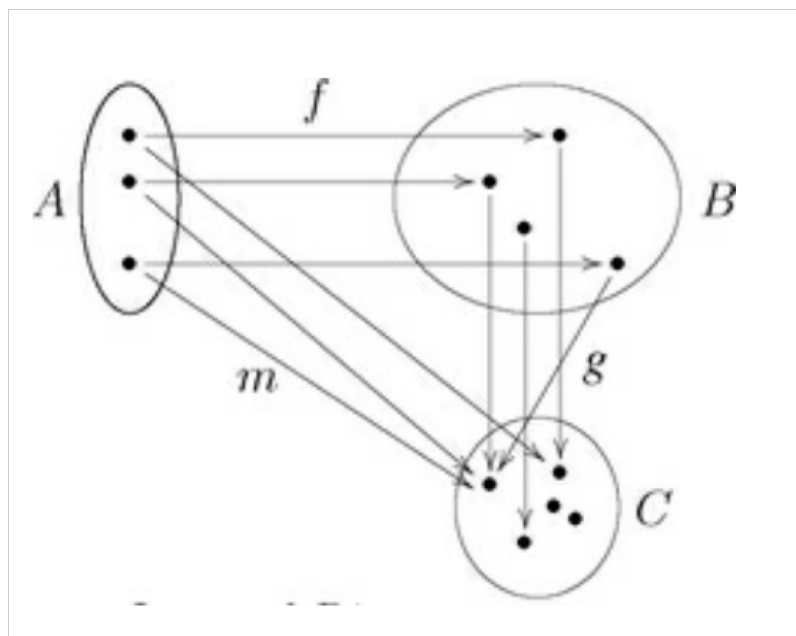
Categoryとは？

- Categoryは、Object(対象)と、Object間の関係を示す Arrow(射:mappingともいう)からなる。
 - 対象Aから 対象Bへの 射 f を、
 $f: A \rightarrow B$ とあらわす。
 - 対象 $A, B, C, \dots$ を集合として、射 $f, g, h, \dots$ を、集合の間に定義された関数としてイメージしてみるといい。Categoryは、それを抽象化したものである。
-

対象(の内部)と射 $f: A \rightarrow B$ のイメージ



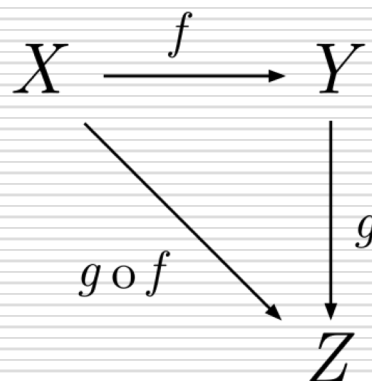
三つの対象 A, B, C と三つの射 f, g, m の例



Categoryの満たすべき条件

- 対象と射の集まりが、すべてCategoryになるわけではない。
Categoryは、次の条件を満たさねばならない。

1. $f: X \rightarrow Y, g: Y \rightarrow Z$ に対して、射の「合成」
 $g \circ f: X \rightarrow Z$ が存在する。



2. 全ての対象 X に対して、恒等射 $1_X: X \rightarrow X$ が存在して、
 $f: X \rightarrow Y$ に対して、 $f \circ 1_X = f = 1_Y \circ f$
-

Functor(関手)とは？

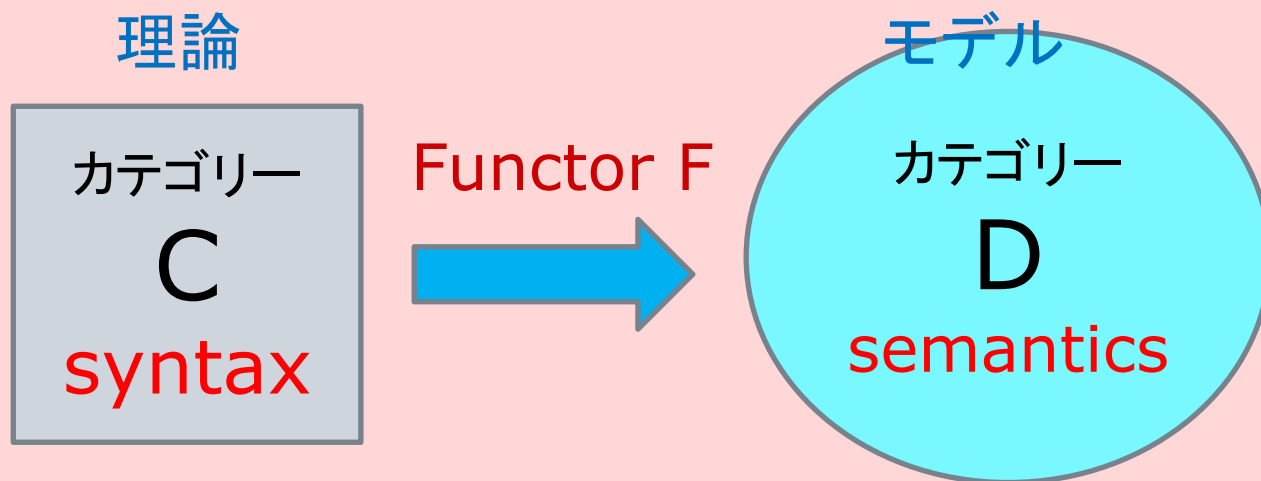
- Category C と Category Dがあった時
 - ・ Cの対象 X を Dの対象 $F(X)$ に
 - ・ Cの射 f を Dの射 $F(f)$ に対応づける F をFunctorと呼ぶ。

 - Cの射 $f: X \rightarrow Y$ を Dの射 $F(f): F(X) \rightarrow F(Y)$ に対応させるものを、Covariant Functor(共変関手)と呼ぶ。

 - Cの射 $f: X \rightarrow Y$ を Dの射 $F(f): F(Y) \rightarrow F(X)$ に対応させるものを、Contravariant Functor(反変関手)と呼ぶ。
-

ローヴェールのFunctor Semantics

- ローベールは、カテゴリーCからカテゴリーDへの、構造を保存するようなFunctor Fは、Dの中でCのある「解釈」を与えることに気づいた。
- それは、Cが記号の結合方法を“Syntax”として記述し、Dは、その“Semantics”を与えるということである。Functor Fは、 $\text{Syntax} \rightarrow \text{Semantics}$ として働き、Syntaxにそれが成り立つ「意味」を持つ世界の「モデル」を与える。



FUNCTORIAL SEMANTICS OF ALGEBRAIC THEORIES

F. WILLIAM LAWVERE

1963年

[https://www.pnas.org/content/pnas/50/5/869.full
.pdf](https://www.pnas.org/content/pnas/50/5/869.full.pdf)

Doctrine

John Baez <https://goo.gl/u63cvK>

Universal algebra began as an attempt to deal with lots of familiar algebraic gadgets simultaneously: groups, rings, lattices, vector spaces, Lie algebras, and so on. A bunch of the same theorems hold for *all* of these - so why not prove them all at once and be done with it? It turns out to be possible!

In his famous 1963 thesis, **Bill Lawvere** showed how universal algebra can be formulated using categories:

F. William Lawvere, ***Functorial Semantics of Algebraic Theories***.

He saw that universal algebra was all about algebraic gadgets that make sense in any category with finite products. Shockingly, every such category $\mathcal{C}$ can be seen as the “theory” of some such gadget. If D is some other category with finite products, a product-preserving functor $F:\mathcal{C}\rightarrow D$: then gives such a gadget in D . We call this a **model** of $\mathcal{C}$ in D .

However, besides categories with finite products, there are also other kinds of categories, which support other kinds of algebraic gadgets. So, around 1969, Lawvere massively generalized universal algebra - so much that he required a precise definition of “a kind of category”! He called such a thing a “doctrine”:

F. William Lawvere, Ordinal sums and equational doctrines, Springer Lecture Notes in Mathematics No. 80, Springer-Verlag (1969), pp. 141-155.

This is a startling leap in abstraction, typical of Lawvere's work.

サンプル – 群の表現

- 次のように定義される数学的な構造 D_3 を考えよう。

$$D_3 = \langle r, s \mid r^3 = s^2 = rsrs = 1 \rangle$$

これだけだと、これがどんな構造かはよくわからない。

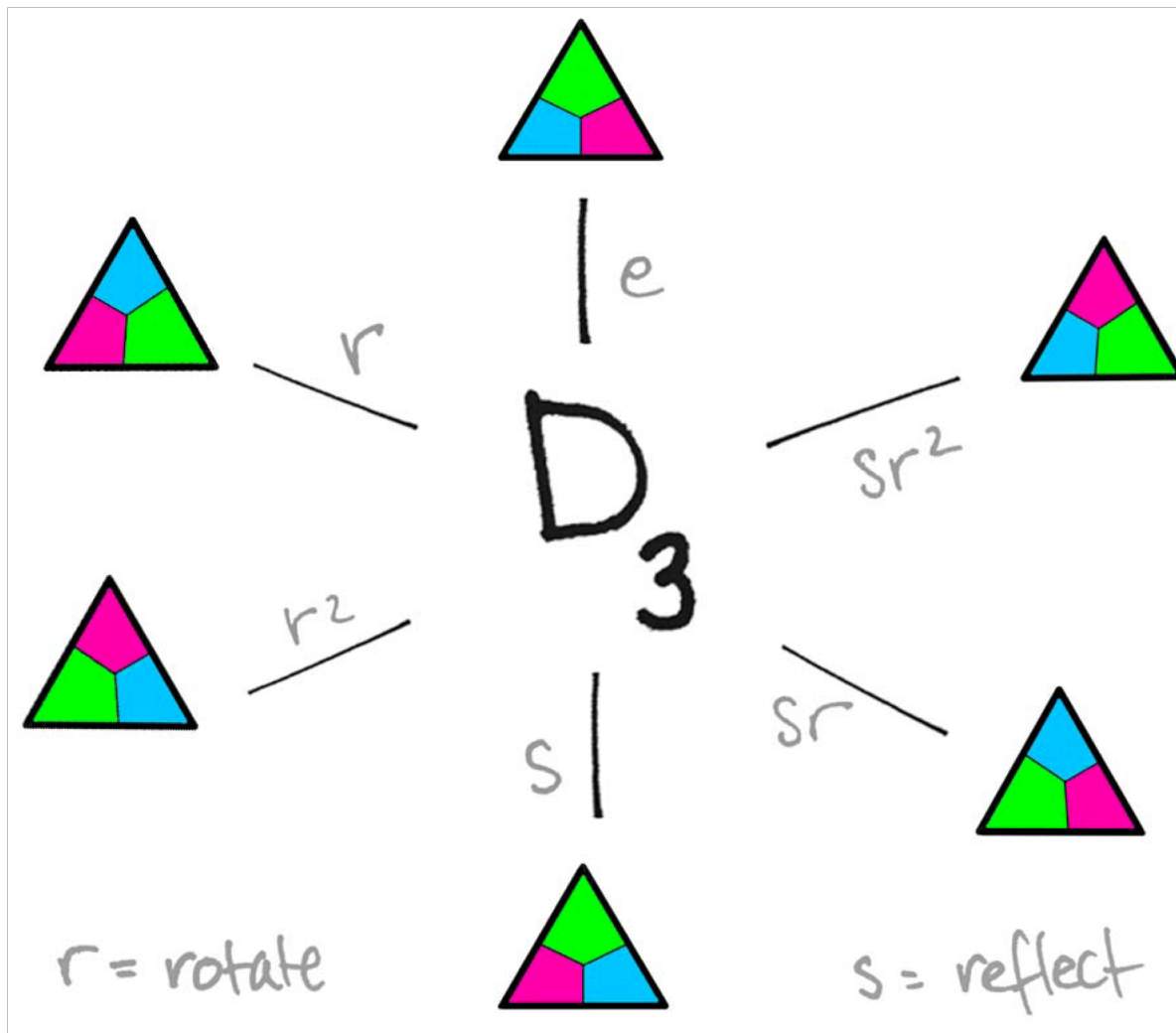
- D_3 の r, s に次のような R と S を対応させる。(Functorだ！)

$$r \mapsto R = \begin{bmatrix} \cos(2\pi/3) & -\sin(2\pi/3) \\ \sin(2\pi/3) & \cos(2\pi/3) \end{bmatrix} \quad s \mapsto S = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

R は、左回りに60度の回転、

S は、垂直軸に沿って180度の回転。裏返しにする。

- これでも、まだ分かりづらいかもしれない。
次の図を見れば、対応ははっきりする。
-



"What is applied category theory?"
<https://arxiv.org/pdf/1809.05923.pdf>

ローベールのコメント

Much of mathematics consists in calculating in various abstract theories, specifically interpreting one abstract theory into another, interpreting an abstract theory into a background to obtain a concrete category of structures, and transforming these structures in and among these categories. Now, **for one thing**, the use of the term “category” of structures of a certain kind had already become obvious in the 1950’s and **for another thing** the idea of theories themselves as structures whose mutual interpretations would form a category was also evidently possible if one cared to carry it out, and indeed Hall, Halmos, Henkin, Tarski, and possibly others had already made significant moves in that direction. **But what of the relation itself between abstract theory and concrete background?**

<https://goo.gl/RLs94Q>

ローベールのコメント

To conceptually relate any two things, it is necessary that they belong to a common category; that is, speaking more mathematically, it is first necessary to functorially transport them into a common third category (if indeed they were initially conceived as belonging to different categories); but then if the attempt to relate them is successful, the clearest exposition of the whole matter may involve presenting the two things themselves as citizens of that third category. In the case of “backgrounds” such as a universe of sets or spaces it was already clear that they form categories, wherein the basic structure is composition. In the case of logical theories of all sorts the most basic structure they support is an operation of substitution, which is most effectively viewed as a form of composition. Thus, if we construe theories as categories, models are functors!

<https://goo.gl/RLs94Q>

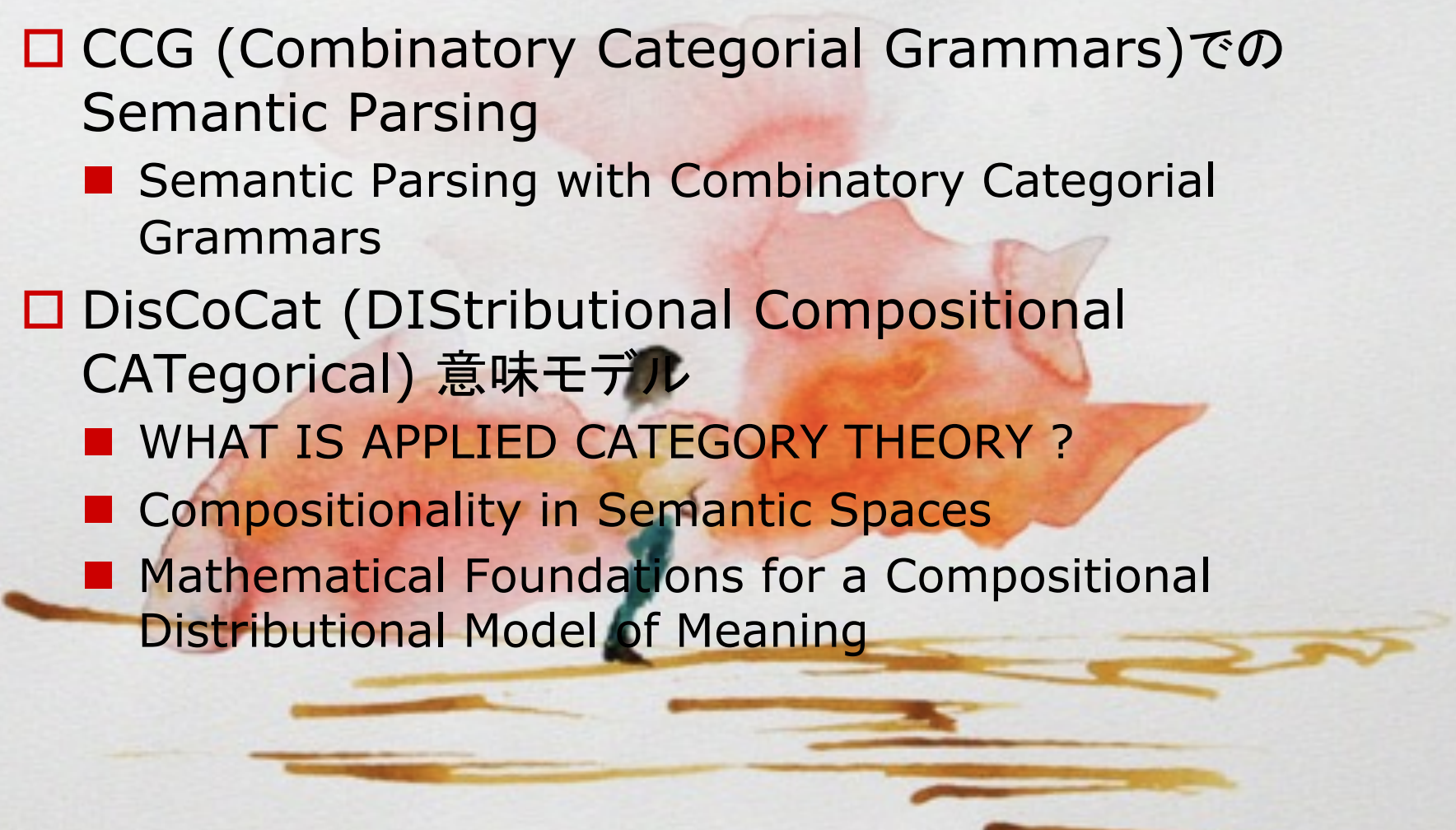
Part III

意味の形式的理論の現在



Part III 意味の形式的理論の現在

Agenda

- CCG (Combinatory Categorical Grammars)での Semantic Parsing
 - Semantic Parsing with Combinatory Categorical Grammars
 - DisCoCat (DIStributIonal Compositional CATegorical) 意味モデル
 - WHAT IS APPLIED CATEGORY THEORY ?
 - Compositionality in Semantic Spaces
 - Mathematical Foundations for a Compositional Distributional Model of Meaning
- 

基本的な観察

文と意味の「構成性(compositionality)」

- 文は文法に従って語から構成される
- 文の意味は、語の意味に依存する
 - 文法的に正しいが意味がない文もある
Colorless green ideas sleep furiously.
- 文の意味は、文の文法的構成に依存する
 - 二つの意味を持つ、あいまいな文
I saw a man with a telescope.

意味の「同一性」 / 意味の共通表現の存在

- 言語が異なっても、「同じ」意味を持つ語が存在する。
- 言語が異なっても、「同じ」意味を持つ文が存在する。

現在の主要な自然言語処理の実装は 文法・文の構成性を考慮していない

	語の意味の 「表現」	文の構成性 Grammar / Syntax	文の意味の 「表現」 Semantics
Intentモデル	X	X	X
Entityモデル	Entityの Ontology	X	Action Typeの Property
機械翻訳モデル	X	X	多次元ベクトル による分散表現 (暗黙の利用)

文法と意味の表現を意識したモデルも存在する

	語の意味の 「表現」	文の構成性 Grammar / Syntax	文の意味の 「表現」 Semantics
Word2Vec	多次元ベクトル による分散表現	—	—
IBM Watson内 のプロジェクト	辞書項目	English Slot Grammar	論理式
Combinatory Categorial Grammar	辞書項目	Categorial Grammar	型付きラムダ式
DisCoCat	多次元ベクトル による分散表現	Pregroup Grammar	多次元ベクトル による分散表現

第三部で紹介

文法と意味の表現を意識したモデルも存在する

	語の意味の 「表現」	文の構成性 Grammar / Syntax	文の意味の 「表現」 Semantics
Word2Vec	多次元ベクトル による分散表現	—	—
IBM Watson内 のプロジェクト	辞書項目	English Slot Grammar	論理式
Combinatory Categorial Grammar	辞書項目	Categorial Grammar Curry—Howard対応	型付きラムダ式
DisCoCat	多次元ベクトル による分散表現	Pregroup Grammar Functor Semantic	多次元ベクトル による分散表現

この第三部で紹介

CCG

(Combinatory Categorical Grammars)での
Semantic Parsing

CCG Semantic Parsing

- CCGでは、語からの文の構成と意味の構成が、同時に、論理的な証明図式の形で行われる。とてもエレガントで強力なアプローチである。
- 語彙項目に、文法的な情報をCategory文法の「型」として、また意味論的な情報を「ラムダ式」として、その双方を持たせることによって、文と意味の構成を同時に行なっている。
- 意味の表現は、ラムダ式で分散表現ではない。
- 理論的な基礎は、Curry-Howard対応である。Curry-Howard対応については、2013年のマルレク「「型の理論」と証明支援システム — Coqの世界」
<https://goo.gl/UMmKSB> を参照してほしい。

Semantic Parsing with Combinatory Categorical Grammars

Yoav Artzi, Nicholas FitzGerald and Luke
Zettlemoyer! : University of Washington

<https://yoavartzi.com/pub/afz-tutorial.acl.2013.pdf>



Language to Meaning



at the chair, move forward three steps past the sofa

$$\lambda a. pre(a, \iota x. chair(x)) \wedge move(a) \wedge len(a, 3) \wedge$$
$$dir(a, forward) \wedge past(a, \iota y. sofa(y))$$

Language to Meaning



at the chair, move forward three steps past the sofa

$$\lambda a. pre(a, \iota x. chair(x)) \wedge move(a) \wedge len(a, 3) \wedge$$
$$dir(a, forward) \wedge past(a, \iota y. sofa(y))$$

Language to Meaning

at the chair, move forward three steps past the sofa

$\lambda a. pre(a, \iota x. chair(x)) \wedge move(a) \wedge len(a, 3) \wedge$
 $dir(a, forward) \wedge past(a, \iota y. sofa(y))$



$f : \text{sentence} \rightarrow \text{logical form}$

Today

Parsing

Combinatory Categorical Grammars

Learning

Unified learning algorithm

Modeling

Best practices for semantics design

UW SPF

Open source semantic parsing framework

<http://yoavartzi.com/spf>

Semantic
Parser

Flexible High-Order
Logic Representation

Learning
Algorithms

Includes ready-to-run examples

Lambda Calculus

Literals

- Represent function application

city(AUSTIN)



Logical expression

List of logical expressions

Lambda Calculus

Lambda Terms

- Bind/scope a variable
- Repeat to bind multiple variables

$$\lambda x.city(x)$$
$$\lambda x.\lambda y.located_in(x, y)$$

Lambda Calculus

Lambda Terms

- Bind/scope a variable
- Repeat to bind multiple variables

$\lambda x.city(x)$

$\lambda x.\lambda y.located_in(x, y)$ Body

Lambda
operator

Variable



Lambda Calculus

Quantifiers?

- Higher order constants
- No need for any special mechanics
- Can represent all of first order logic

$\forall(\lambda x.big(x) \wedge apple(x))$

$\neg(\exists(\lambda x.lovely(x))$

$\iota(\lambda x.beautiful(x) \wedge grammar(x))$

Simply Typed Lambda Calculus

$\lambda a. \text{move}(a) \wedge \text{dir}(a, \text{LEFT}) \wedge \text{to}(a, \iota y. \text{chair}(y)) \wedge$
 $\text{pass}(a, \lambda y. \text{sofa}(y) \wedge \text{intersect}(\lambda z. \text{intersection}(z), y))$

Type information usually omitted

Capturing Meaning with Lambda Calculus

SYSTEM how can I help you ?

USER i ' d like to fly to new york

SYSTEM flying to new york . leaving what city ?

USER from boston on june seven with american airlines

SYSTEM flying to new york . what date would you like to depart boston ?

USER june seventh

SYSTEM do you have a preferred airline ?

USER american airlines

SYSTEM o . k . leaving boston to new york on june seventh flying with
american airlines . where would you like to go to next ?

USER back to boston on june tenth

[CONVERSATION CONTINUES]

Combinatory Categorical Grammars

$$\begin{array}{c}
 \text{CCG} \qquad \text{is} \qquad \text{fun} \\
 \hline
 \text{NP} \qquad S \backslash NP / ADJ \qquad ADJ \\
 \text{CCG} \qquad \lambda f. \lambda x. f(x) \qquad \lambda x. fun(x) \\
 \hline
 \qquad \qquad S \backslash NP \\
 \qquad \qquad \lambda x. fun(x) \\
 \hline
 \qquad \qquad S \\
 \qquad \qquad fun(CCG)
 \end{array}$$

CCG Categories

$ADJ : \lambda x.fun(x)$

- Basic building block
- Capture syntactic and semantic information jointly

CCG Categories

Syntax

ADJ

$\lambda x. fun(x)$

Semantics

- Basic building block
- Capture syntactic and semantic information jointly

CCG Categories

Syntax

$ADJ : \lambda x. fun(x)$

$(S \backslash NP) / ADJ : \lambda f. \lambda x. f(x)$

$NP : CCG$

- Primitive symbols: N, S, NP, ADJ and PP
- Syntactic combination operator (/,\)
- Slashes specify argument order and direction

CCG Categories

$ADJ : \lambda x. fun(x)$ Semantics

$(S \setminus NP) / ADJ : \lambda f. \lambda x. f(x)$

$NP : CCG$

- λ -calculus expression
- Syntactic type maps to semantic type

CCG Lexical Entries

$\text{fun} \vdash ADJ : \lambda x. fun(x)$

- Pair words and phrases with meaning
- Meaning captured by a CCG category

CCG Lexical Entries



- Pair words and phrases with meaning
- Meaning captured by a CCG category

CCG Lexicons

$\text{fun} \vdash ADJ : \lambda x. fun(x)$

$\text{is} \vdash (S \backslash NP) / ADJ : \lambda f. \lambda x. f(x)$

$CCG \vdash NP : CCG$

- Pair words and phrases with meaning
- Meaning captured by a CCG category

CCG Operations

- Small set of operators
 - Input: 1-2 CCG categories
 - Output: A single CCG category
- Operate on syntax semantics together
- Mirror natural logic operations

CCG Operations

Application

$$B : g \quad A \backslash B : f \Rightarrow A : f(g) \quad (<)$$

$$A / B : f \quad B : g \Rightarrow A : f(g) \quad (>)$$

- Equivalent to function application
- Two directions: forward and backward
 - Determined by slash direction

“CCG is fun”

Parsing with CCGs

CCG	is	fun
$\overline{NP}$ CCG	$\overline{S \backslash NP / ADJ}$ $\lambda f. \lambda x. f(x)$	$\overline{ADJ}$ $\lambda x. fun(x)$
	$\overline{\hspace{10em}} >$ $S \backslash NP$ $\lambda x. fun(x)$	

Combine categories using operators

$$A/B : f \quad B : g \Rightarrow A : f(g) \quad (>)$$

“CCG is fun”

Parsing with CCGs

CCG	is	fun
$\overline{NP}$ $\overline{CCG}$	$\overline{S \backslash NP / ADJ}$ $\lambda f. \lambda x. f(x)$	$\overline{ADJ}$ $\lambda x. fun(x)$
	$\overline{\hspace{10em}} >$	
	$S \backslash NP$ $\lambda x. fun(x)$	
	$\overline{\hspace{10em}} <$	
	S $fun(CCG)$	

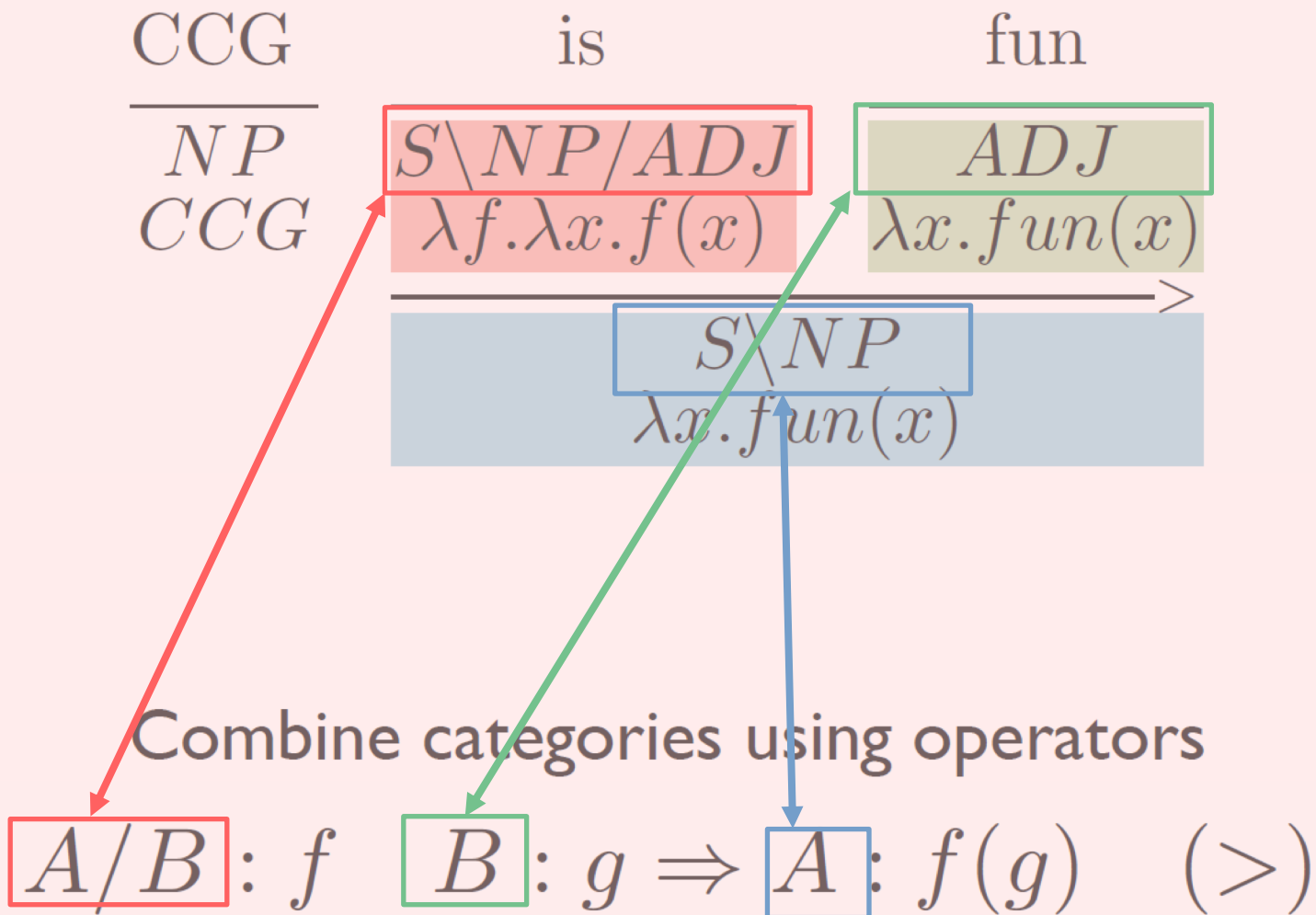
論理式での表現

Combine categories using operators

$$B : g \quad A \backslash B : f \Rightarrow A : f(g) \quad (<)$$

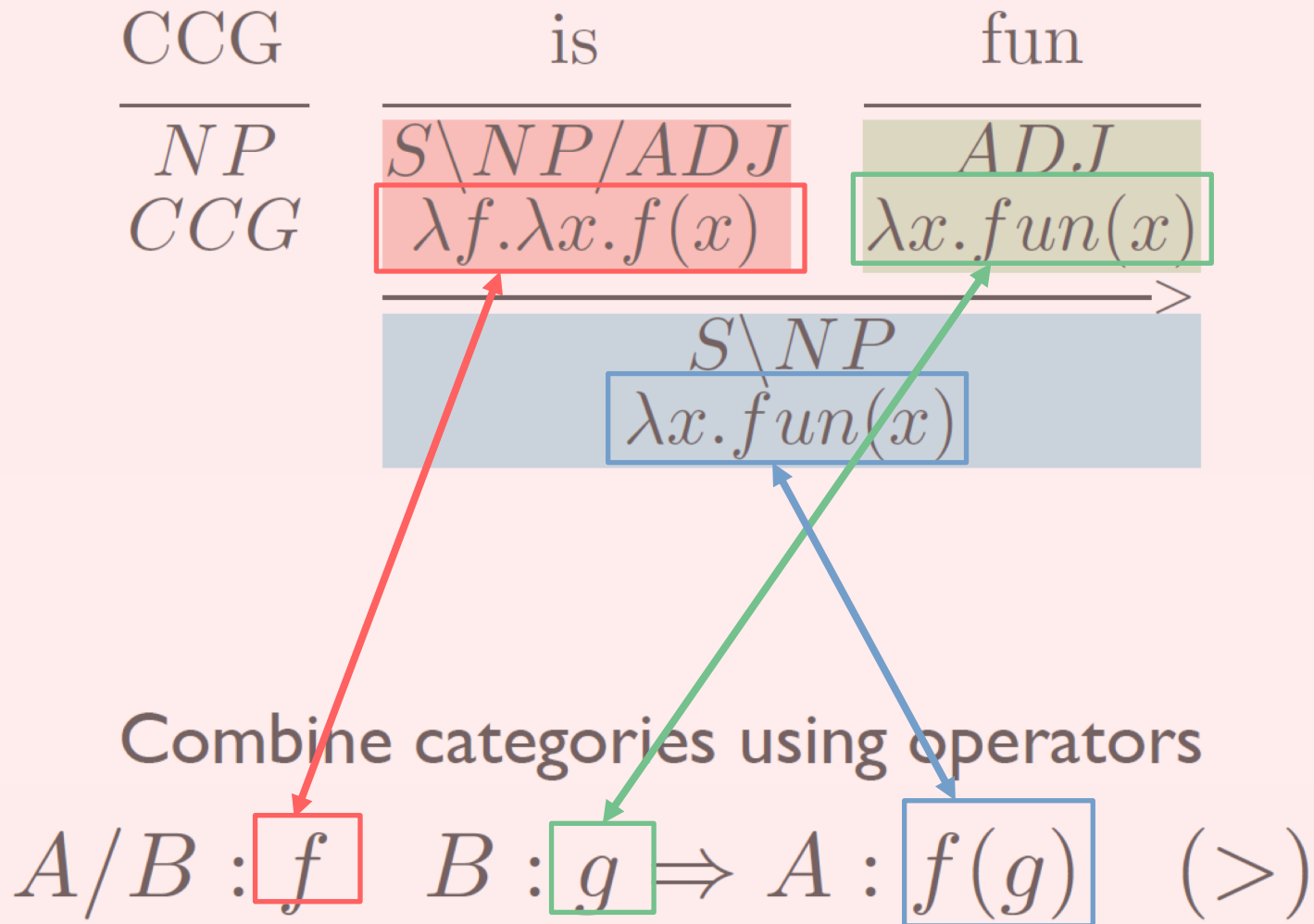
“CCG is fun”

Parsing with CCGs



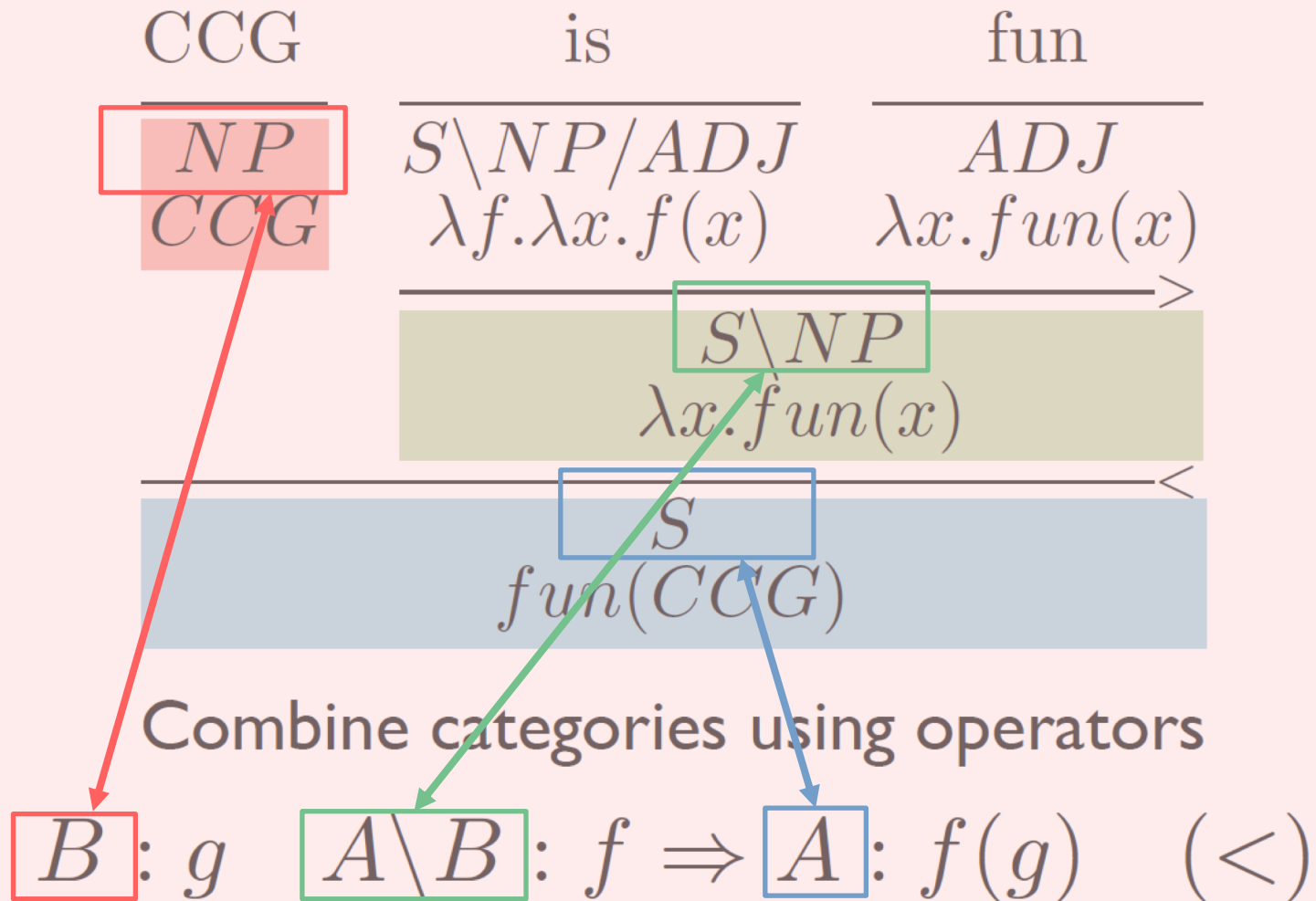
“CCG is fun”

Parsing with CCGs



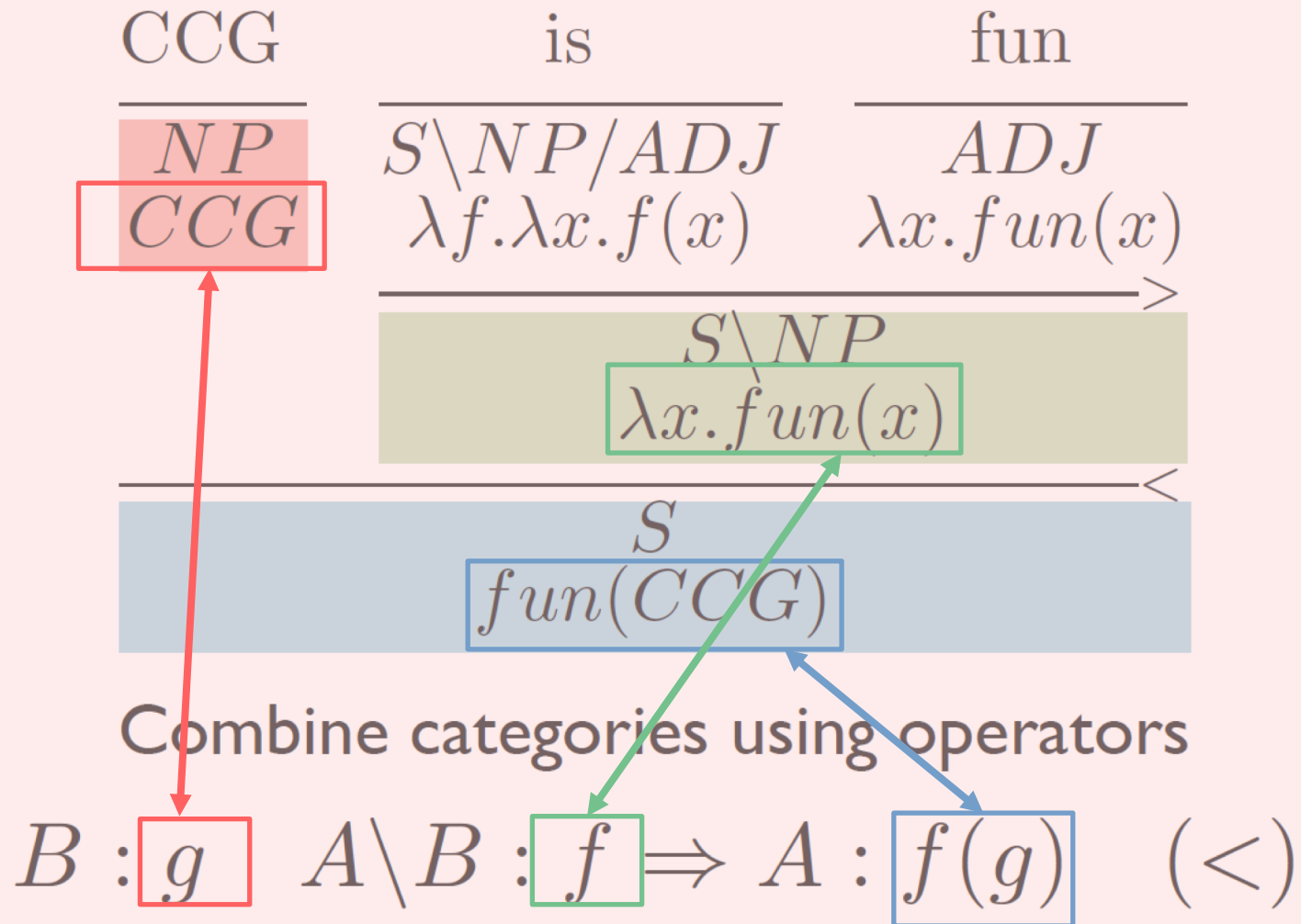
“CCG is fun”

Parsing with CCGs



“CCG is fun”

Parsing with CCGs



CCG Operations

Composition

$$A/B : f \quad B/C : g \Rightarrow A/C : \lambda x.f(g(x)) \quad (> B)$$

$$B \backslash C : g \quad A \backslash B : f \Rightarrow A \backslash C : \lambda x.f(g(x)) \quad (< B)$$

- Equivalent to function composition*
- Two directions: forward and backward

* Formal definition of logical composition in supplementary slides

CCG Operations

Composition

$$\begin{array}{l} \boxed{\begin{array}{c} f \\ A/B : f \end{array}} \quad \boxed{\begin{array}{c} g \\ B/C : g \end{array}} \Rightarrow \boxed{\begin{array}{c} f \circ g \\ A/C : \lambda x. f(g(x)) \end{array}} \quad (> B) \\ B \setminus C : g \quad A \setminus B : f \Rightarrow A \setminus C : \lambda x. f(g(x)) \quad (< B) \end{array}$$

- Equivalent to function composition*
- Two directions: forward and backward

* Formal definition of logical composition in supplementary slides

CCG Operations

Type Shifting

$$ADJ : \lambda x.g(x) \Rightarrow N/N : \lambda f.\lambda x.f(x) \wedge g(x)$$

$$PP : \lambda x.g(x) \Rightarrow N \setminus N : \lambda f.\lambda x.f(x) \wedge g(x)$$

$$AP : \lambda e.g(e) \Rightarrow S \setminus S : \lambda f.\lambda e.f(e) \wedge g(e)$$

$$AP : \lambda e.g(e) \Rightarrow S/S : \lambda f.\lambda e.f(e) \wedge g(e)$$

- Category-specific unary operations
- Modify category type to take an argument
- Helps in keeping a compact lexicon

CCG Operations

Type Shifting

Input	Output
$ADJ : \lambda x.g(x)$	$\Rightarrow N/N : \lambda f.\lambda x.f(x) \wedge g(x)$
$PP : \lambda x.g(x)$	$\Rightarrow N \setminus N : \lambda f.\lambda x.f(x) \wedge g(x)$
$AP : \lambda e.g(e)$	$\Rightarrow S \setminus S : \lambda f.\lambda e.f(e) \wedge g(e)$
$AP : \lambda e.g(e)$	$\Rightarrow S/S : \lambda f.\lambda e.f(e) \wedge g(e)$

- Category-specific unary operations
- Modify category type to take an argument
- Helps in keeping a compact lexicon

CCG Operations

Coordination

and $\vdash C : conj$

or $\vdash C : disj$

- Coordination is special cased
 - Specific rules perform coordination
 - Coordinating operators are marked with special lexical entries

“square blue or round yellow pillow”

Parsing with CCGs

square	blue	or	round	yellow	pillow
<i>ADJ</i>	<i>ADJ</i>	<i>C</i>	<i>ADJ</i>	<i>ADJ</i>	<i>N</i>
$\lambda x.square(x)$	$\lambda x.blue(x)$	<i>disj</i>	$\lambda x.round(x)$	$\lambda x.yellow(x)$	$\lambda x.pillow(x)$

Use lexicon to match words and phrases with their categories

“square blue or round yellow pillow”

Parsing with CCGs

square	blue	or	round	yellow	pillow
ADJ	ADJ	C	ADJ	ADJ	N
$\lambda x.square(x)$	$\lambda x.blue(x)$	$disj$	$\lambda x.round(x)$	$\lambda x.yellow(x)$	$\lambda x.pillow(x)$
N/N					
$\lambda f.\lambda x.f(x) \wedge square(x)$					

Shift adjectives to combine

$$ADJ : \lambda x.g(x) \Rightarrow N/N : \lambda f.\lambda x.f(x) \wedge g(x)$$

“square blue or round yellow pillow”

Parsing with CCGs

square	blue	or	round	yellow	pillow
$\overline{ADJ}$ $\lambda x.square(x)$	$\overline{ADJ}$ $\lambda x.blue(x)$	$\overline{C}$ $disj$	$\overline{ADJ}$ $\lambda x.round(x)$	$\overline{ADJ}$ $\lambda x.yellow(x)$	$\overline{N}$ $\lambda x.pillow(x)$
$\overline{N/N}$ $\lambda f.\lambda x.f(x) \wedge square(x)$	$\overline{N/N}$ $\lambda f.\lambda x.f(x) \wedge blue(x)$		$\overline{N/N}$ $\lambda f.\lambda x.f(x) \wedge round(x)$	$\overline{N/N}$ $\lambda f.\lambda x.f(x) \wedge yellow(x)$	

Shift adjectives to combine

$$ADJ : \lambda x.g(x) \Rightarrow N/N : \lambda f.\lambda x.f(x) \wedge g(x)$$

“square blue or round yellow pillow”

Parsing with CCGs

square	blue	or	round	yellow	pillow
$\frac{}{ADJ}$	$\frac{}{ADJ}$	$\frac{}{C}$	$\frac{}{ADJ}$	$\frac{}{ADJ}$	$\frac{}{N}$
$\lambda x.square(x)$	$\lambda x.blue(x)$	$disj$	$\lambda x.round(x)$	$\lambda x.yellow(x)$	$\lambda x.pillow(x)$
$\frac{}{N/N}$	$\frac{}{N/N}$		$\frac{}{N/N}$	$\frac{}{N/N}$	
$\lambda f.\lambda x.f(x) \wedge square(x)$	$\lambda f.\lambda x.f(x) \wedge blue(x)$		$\lambda f.\lambda x.f(x) \wedge round(x)$	$\lambda f.\lambda x.f(x) \wedge yellow(x)$	
$\xrightarrow{>B}$			$\xrightarrow{>B}$		
$\frac{}{N/N}$			$\frac{}{N/N}$		
$\lambda f.\lambda x.f(x) \wedge square(x) \wedge blue(x)$			$\lambda f.\lambda x.f(x) \wedge round(x) \wedge yellow(x)$		

Compose pairs of adjectives

$$A/B : f \quad B/C : g \Rightarrow A/C : \lambda x.f(g(x)) \quad (> B)$$

“square blue or round yellow pillow”

Parsing with CCGs

square	blue	or	round	yellow	pillow
ADJ $\lambda x.square(x)$	ADJ $\lambda x.blue(x)$	C $disj$	ADJ $\lambda x.round(x)$	ADJ $\lambda x.yellow(x)$	N $\lambda x.pillow(x)$
N/N $\lambda f.\lambda x.f(x) \wedge square(x)$	N/N $\lambda f.\lambda x.f(x) \wedge blue(x)$		N/N $\lambda f.\lambda x.f(x) \wedge round(x)$	N/N $\lambda f.\lambda x.f(x) \wedge yellow(x)$	
N/N $\lambda f.\lambda x.f(x) \wedge square(x) \wedge blue(x)$			N/N $\lambda f.\lambda x.f(x) \wedge round(x) \wedge yellow(x)$		
N/N $\lambda f.\lambda x.f(x) \wedge ((square(x) \wedge blue(x)) \vee (round(x) \wedge yellow(x)))$					

Coordinate composed adjectives

“square blue or round yellow pillow”

Parsing with CCGs

square	blue	or	round	yellow	pillow
ADJ $\lambda x.square(x)$	ADJ $\lambda x.blue(x)$	C $disj$	ADJ $\lambda x.round(x)$	ADJ $\lambda x.yellow(x)$	N $\lambda x.pillow(x)$
N/N $\lambda f.\lambda x.f(x) \wedge square(x)$	N/N $\lambda f.\lambda x.f(x) \wedge blue(x)$		N/N $\lambda f.\lambda x.f(x) \wedge round(x)$	N/N $\lambda f.\lambda x.f(x) \wedge yellow(x)$	
N/N $\lambda f.\lambda x.f(x) \wedge square(x) \wedge blue(x)$			N/N $\lambda f.\lambda x.f(x) \wedge round(x) \wedge yellow(x)$		
					$<\Phi>$
N/N $\lambda f.\lambda x.f(x) \wedge ((square(x) \wedge blue(x)) \vee (round(x) \wedge yellow(x)))$					
N $\lambda x.pillow(x) \wedge ((square(x) \wedge blue(x)) \vee (round(x) \wedge yellow(x)))$					

Apply coordinated adjectives to noun

$$A/B : f \quad B : g \Rightarrow A : f(g) \quad (>)$$

DisCoCat

DIStributional COMpositional CATegorical

意味モデル

Category論とFunctor Semanticsの
言語理論への応用

WHAT IS APPLIED CATEGORY THEORY ?

Tai-Danae Bradley

2018年

[https://arxiv.org/pdf/
1809.05923.pdf](https://arxiv.org/pdf/1809.05923.pdf)



Syntax = 文法

Semantics = 意味

- モデル論的なSyntaxとSemanticsの二項性と双対性の認識は、言語の意味を考える場合、非常に示唆的である。
- 文が語から文法に従って構成され、文の意味もそれを構成する語の意味から構成されるのなら、ほとんど同語反復なのだが、次のように考えることができる。

コンピュータが個々の語の意味を理解し、文法も理解できるなら、その両方を使ってコンピュータに、文全体の意味を理解させることはできないか？

意味の形式的理論

□ 「Syntax = 文法」と「Semantics = 意味」の対応づけに、Functor Semanticsのフレームを使うには、どのような準備が必要だろうか？ 次の三つが考えられる。

1. **文法のCategory**: 文法を数学的に扱うにはどうすればよいのか？ 文法を数学的なCategoryとして捉える。
2. **意味のCategory**: 意味を数学的に扱うにはどうすればよいのか？ 意味を数学的なCategoryとして捉える。
3. **Functor**: 文法 $\rightarrow$ 意味 のFunctorをどのように構成するか？ このFunctorは、文全体の意味を捉えることができるか？

文法のCategory

- 文法のCategoryでは、前々回のマルレクで紹介した Lambekの方法が有力である。
- 基本的な文法の型は、名詞の型 n と文の型 s である。これから、基本的な品詞の型が定義されていく。

n = 名詞

s = 文

nn^l = 形容詞

$n^r s n^l$ = 他動詞

- 語が接続する時、対応する語の型で、次のような計算を行う。

$ww^l = \varepsilon^l$ (消える)

$w^r w = \varepsilon^r$ (消える)

"yellow banana" は名詞として扱われる

"yellow" is
an adjective



yellow

n n^e

apply the
identity on n
here

"banana" is
a noun



banana

n

apply the
counit ε^e here

$$1_n \cdot \varepsilon^e$$

→

"yellow banana"
is a noun



yellow banana

$$1 \cdot n = n$$

$n \cdot 1$

Success!

"bananas are fruit" は文である

"bananas" is
a noun



bananas

"are" is
a verb

are

"fruit" is
a noun



fruit

"bananas are fruit"
is a sentence



bananas are fruit

n n^r s n^l n
 └───┬───┘ └──┬──┘ └───┬───┘
 apply ϵ^r apply apply ϵ^l
 identity

$\epsilon^r \cdot 1_s \cdot \epsilon^l$
 ───────────→

s

$$nn^r sn^l n = nn^r \cdot s \cdot n^l n \xrightarrow{\epsilon^r \cdot 1_s \cdot 1_n} 1 \cdot s \cdot n^l n = s \cdot n^l n \xrightarrow{1_s \cdot \epsilon^l} s \cdot 1 = s$$

意味の表現について

- 意味の表現については、二つの流儀がある。
 - 論理的な形式を用いるもの
 - ベクトル空間で表現するもの

- 後者の流儀の基本的な論文は、次のものである。

Coecke, B., Sadrzadeh, M., & Clark, S. (2011).
"Mathematical Foundations for a Compositional
Distributional Model of Meaning."
Linguistic Analysis, 36(1-4), 345-384.
<https://arxiv.org/abs/1003.4394>

Functor: syntax $\rightarrow$ semantics

- To summarize, the language model of Coecke et. al. amounts to a structure-preserving functor
syntax $\rightarrow$ semantics
- In this section, we've just shown that the syntax category is taken to be a pregroup freely generated on a finite set of basic grammar types, i.e. syntax " PregX. Let's move on to semantics now.

The Semantics Category: Vector Spaces

The Semantics Category: Vector Spaces

- As stated in the introduction to Section 3.2, computers are able to understand meanings of individual words pretty well. That's because computers understand numbers! For example, a great way to inform a computer of the meaning of the word banana is to represent banana by a number and then give that number to the computer.
- What number? Well, it's not exactly a number. It's an array of numbers—a vector. Okay, what vector? The answer is simple, though I'd like to motivate it by sharing the following theorem.

語の意味の分散モデル

- whereas puppy occurs more often near words such as pet, cute, furry, bark, and so on. As another example, you might not know what the word yegg means (or perhaps you do), but you can probably infer it from this sentence:
- So we can represent the meaning of a word by a vector. This is often called a distributional model of meaning. But what, exactly, is the assignment word $\mapsto$ vector? Suppose we have a fixed corpus—your favorite book, say. Start by choosing a set of so-called **context words** $\{w_1, \dots, w_n\}$.

語の意味の分散モデル

- This can be every word in the corpus or some subset of it. By representing each w_i as the i th standard basis vector we obtain a basis $\{w_1, \dots, w_n\}$ for a vector space V . Then any word w in the corpus has a vector representation given by a linear combination of the context words

$$\mathbf{w} = \sum_{i=1}^n c_i \mathbf{w}_i$$

- The coefficients c_i are real numbers that indicate the number of times that w occurs near w_i in the corpus

語の意味の分散モデル

- Here's an example. Suppose we're reading a book that contains the words {sweet, green, furry}. Let's choose them to be our context words and make the assignment so that

$$\text{sweet} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad \text{green} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \quad \text{furry} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

$$\text{banana} = \begin{bmatrix} 21 \\ 9 \\ 0 \end{bmatrix} \quad \text{puppy} = \begin{bmatrix} 8 \\ 1 \\ 32 \end{bmatrix} \quad \text{fruit} = \begin{bmatrix} 43 \\ 19 \\ 0 \end{bmatrix}$$

- In other words, we've used data from the corpus to embed these words as vectors inside of a three-dimensional vector space. This prompts us to say that the meaning of the word banana is the vector $(21, 9, 0)$, the meaning of puppy is $(8, 1, 32)$, and the meaning of fruit is $(43, 19, 0)$



- And this works! That is, you can feed distributional models into your computer, and they'll ace the word-similarity portion of your SAT exam. Or you can compute the dot product between words, and you'll find that vectors are closer together precisely when the words they represent have the same meaning. It's all familiar territory for NLP practitioners. The semantics category for Coecke et. al. is thus the category of finite dimensional vector spaces over $\mathbb{R}$. That is,

meanings of words = $\mathbf{FVect}$

Functor: syntax $\rightarrow$ semantics

- Unfortunately, the distributional model does not work for sentences. The same sentence rarely occurs twice in a given document, therefore we can't follow the same procedure above. This is where category theory can help lend a hand. In light of the principle of compositionality, the meaning of a sentence should be able to be computed given the meanings of its individual words and the rules of grammar for combining them. And we can pair meanings-of-words with grammatical types via a map from syntax (grammar) to semantics (meanings of words), i.e. via a functor

$$F : \text{PregX} \rightarrow \text{FVect}$$

- where X is a finite set of basic grammar types. In fact, as we know from Section 2.1, both $\mathbf{FVect}$ and $\mathbf{PregX}$ are compact closed categories, so we'll ask that F be a strong monoidal functor—one that preserves the compact closed structure. That's the gist behind the categorical compositional distributional model of Coecke et. al. But how is F actually defined? Let's talk about that next.

The Functor: Syntax $\rightarrow$ Semantics

- In this section, we'll give an explicit description of the functor $F : \text{syntax} \rightarrow \text{semantics}$ or more specifically, $F : \text{PregX} \rightarrow \text{FVect}$ For simplicity, let's take $X = \{n, s\}$ as we did before. Now to define a functor, we need simply to say what it does on objects and morphisms. So let's do that

The Functor: Syntax $\rightarrow$ Semantics

On objects

- F assigns to the noun type n a vector space $N : F_n$, which we'll call a **noun space**
- F assigns to the sentence type s a vector space $S : F_s$, which we'll call a **sentence space**

On morphisms

- F assigns to a type reduction $a \xrightarrow{r} b$ a linear map $Fa \xrightarrow{Fr} Fb$ that sends the vector corresponding to a word or phrase of type a in Fa to the vector corresponding to a word or phrase of type b in Fb .

The Functor: Syntax $\rightarrow$ Semantics

□ 'F preserve the compact closed structure' means that

- units and counits in $\text{Preg}\{n, s\}$ map to units and counits in FVect

e.g. given $n \in \text{Preg}\{n, s\}$,

$$F(1 \xrightarrow{\eta_n^r} n^r n) = F(1 \xrightarrow{\eta_n^l} n n^l) = \eta_N \quad \text{and} \quad F(n n^r \xrightarrow{\epsilon_n^r} 1) = F(n^l n \xrightarrow{\epsilon_n^l} 1) = \epsilon_N$$

where $\eta_N: \mathbb{R} \rightarrow N \otimes N$ and $\epsilon_N: N \otimes N \rightarrow \mathbb{R}$ are the linear maps that we defined on p. 17. A similar idea holds if we replace n by any element of $\text{Preg}\{n, s\}$.

- duals map to duals

e.g. $Fn^r = Fn^l = N^*$. But our vector spaces are finite dimensional and so $N^* \cong N$ and therefore $Fn^r = Fn^l = N$.

- a compound type is assigned to a tensor product of vector spaces.

$$\text{e.g. } F(n^r s n^l) \cong Fn^r \otimes Fs \otimes Fn^l = N \otimes S \otimes N$$

Compute the meaning of
"bananas are fruit"

Step 1: Assign each word a grammar type
in $\text{Preg}(n, s)$.

banana $\rightarrow n$

are $\rightarrow n^r s n^l$

fruit $\rightarrow n$

Step 2: Fix a **noun space** $F_n = N$ and a sentence space $F_s = S$

Let's suppose N is the three-dimensional space spanned by the basis vectors

sweet , green , furry

which we can represent as column vectors

$$\mathbf{w}_1 = \text{sweet} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{w}_2 = \text{green} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \quad \mathbf{w}_3 = \text{furry} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

These basis vectors generate the noun space.

Step 2: Fix a noun space $F_n = N$ and a sentence space $F_s = S$

For simplicity, let's define S to be a "true or false" space so that it's a one-dimensional vector space spanned by a single vector $\vec{1}$.

The origin $0 \in S$ corresponds to "false" while $\vec{1}$ corresponds to "true."

What about scalar multiples of $\vec{1}$? If you like, you're more than welcome to think of a positive scalar multiple of $\vec{1}$ as the meaning vector for sentence that is super true. The larger the scalar, the more true the sentence!

Finally, note that once we've established N and S , the verb space comes for free:

$$F(n^r s n^l) = N \otimes S \otimes N$$

Step 3: Determine the vector representations of each word in the sentence.

$$\mathbf{bananas} = \begin{bmatrix} 21 \\ 9 \\ 0 \end{bmatrix}, \quad \mathbf{fruit} = \begin{bmatrix} 43 \\ 19 \\ 0 \end{bmatrix}$$

Note that both of these vectors live in the noun space N since each word has grammar type n .

By Step 1, we know that $\mathbf{are}$ has grammar type $n^r s n^l$ and is therefore a vector in the tensor product $N \otimes S \otimes N$.

- And that is excellent news, for if we know “what is what,” i.e. if we know that puppies are furry but not green and so on, then we can re-express the vector for **are** as a 3 x 3 matrix. The ij th entry of this matrix is the coefficient c_{ij} which is

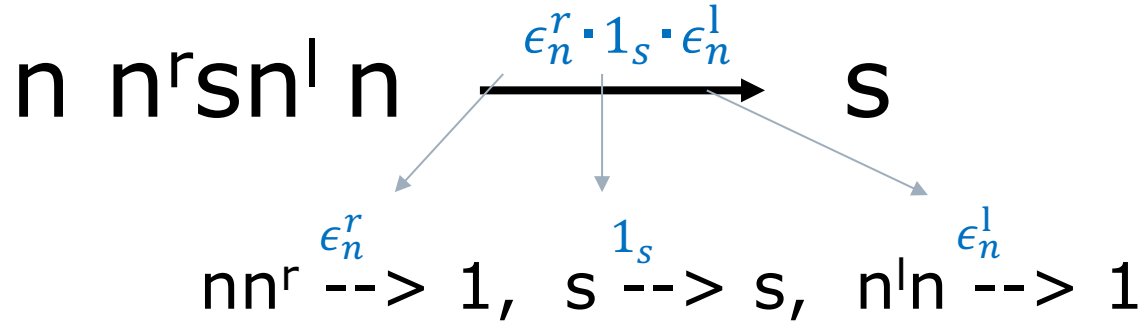
$$c_{ij} = \begin{cases} 1, & \text{if } w_i \text{ is } w_j, \\ 0, & \text{otherwise} \end{cases}$$

?

$$\mathbf{are} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Step 4: Choose a type reduction
in $\text{Preg}(n, s)$

banana is fruit



Step 5: Apply F!

banana is fruit

$$n \ n^r s n^l \ n \xrightarrow{\epsilon_n^r \cdot 1_s \cdot \epsilon_n^l} S$$



$$Fn = N, \ Fn^r = N, \ Fs = S, \ Fn^l = N, \ Fn = N$$

$$F(nn^r \xrightarrow{\epsilon_n^r} 1) = F(n^l n \xrightarrow{\epsilon_n^l} 1) = \epsilon_N$$

$$N \otimes N \otimes S \otimes N \otimes N \xrightarrow{\epsilon_N \otimes 1_S \otimes \epsilon_N} S$$

Functor F

Syntax $\rightarrow$ Semantics

$$\epsilon_N \otimes 1_S \otimes \epsilon_N(\mathbf{bananas} \otimes \mathbf{are} \otimes \mathbf{fruit}) = \begin{bmatrix} 21 & 9 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 43 \\ 19 \\ 0 \end{bmatrix} = 1074$$

The meaning of the sentence *bananas are fruit* is ?

$$1074 \vec{1}$$

$$\Psi = \sum_{ijk} c_{ijk} \vec{v}_i \otimes \vec{s}_j \otimes \vec{w}_k \in V \otimes S \otimes W$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{aligned} f(\vec{v} \otimes \vec{\Psi} \otimes \vec{w}) &= \epsilon_V \otimes 1_S \otimes \epsilon_W(\vec{v} \otimes \vec{\Psi} \otimes \vec{w}) \\ &= \sum_{ijk} c_{ijk} \langle \vec{v} | \vec{v}_i \rangle \vec{s}_j \langle \vec{w}_k | \vec{w} \rangle \\ &= \sum_j \left(\sum_{ik} c_{ijk} \langle \vec{v} | \vec{v}_i \rangle \langle \vec{w}_k | \vec{w} \rangle \right) \vec{s}_j \end{aligned}$$

DisCoCat (DISTRIBUTIONAL COMPOSITIONAL CATEGORICAL) 意味モデル Maaïke Zwart



<https://www.youtube.com/watch?v=99keybEZN4g>

Mathematical Foundations for a Compositional Distributional Model of Meaning

Coecke, B.,
Sadrzadeh,
M., & Clark, S.
2011年
Linguistic Analysis,
36(1-4), 345-384.
[https://arxiv.org/
abs/1003.4394](https://arxiv.org/abs/1003.4394)



□ Basic Type

n : noun

j : infinitive of the verb

s : declarative statement

σ : glueing type

John	likes	Mary
n	$(n^r s n^l)$	n

$$n(n^r s n^l)n \rightarrow 1s n^l n \rightarrow 1s1 \rightarrow s$$

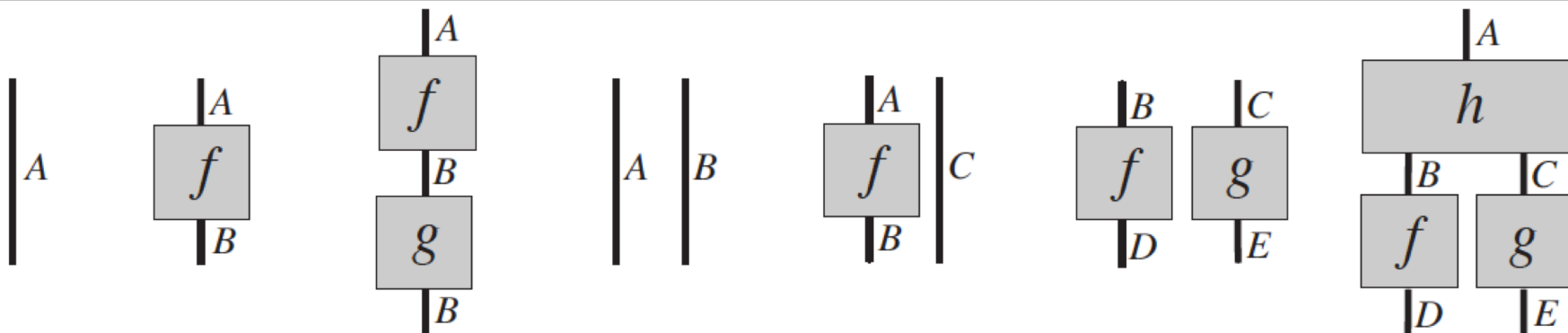
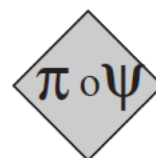
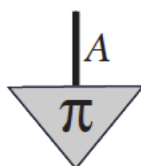
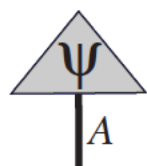
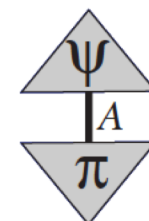
n	n^r	s	n^l	n
				

John	does	not	like	Mary
n	$(n^r s j^l \sigma)$	$(\sigma^r j j^l \sigma)$	$(\sigma^r j n^l)$	n

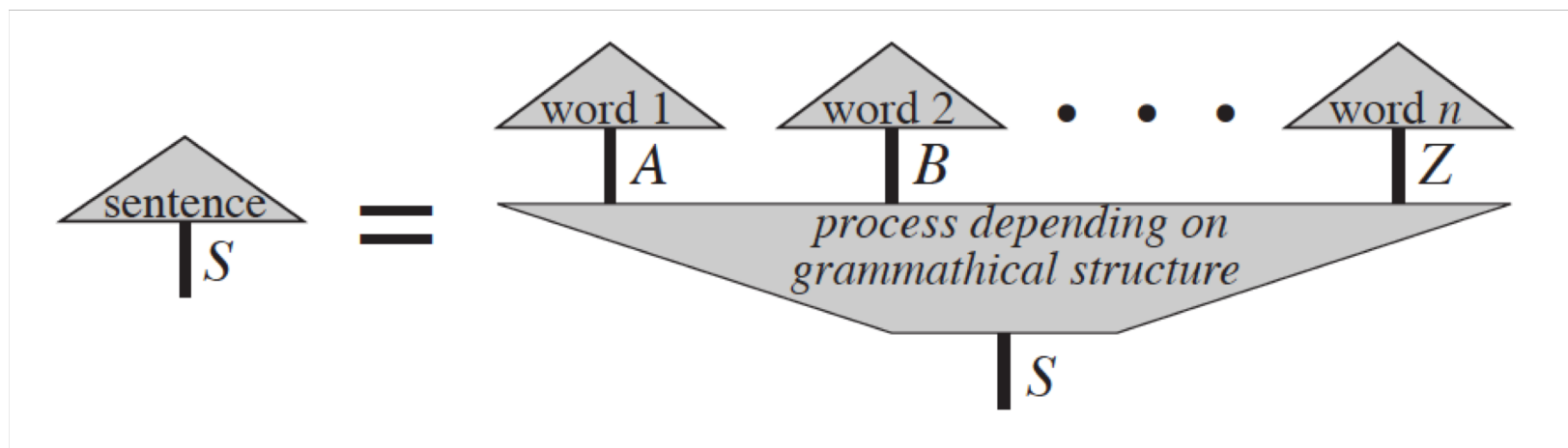
$$n (n^r s j^l \sigma) (\sigma^r j j^l \sigma) (\sigma^r j n^l) n \rightarrow s$$

$$n \quad n^r s j^l \sigma \quad \sigma^r j j^l \sigma \quad \sigma^r j n^l \quad n$$

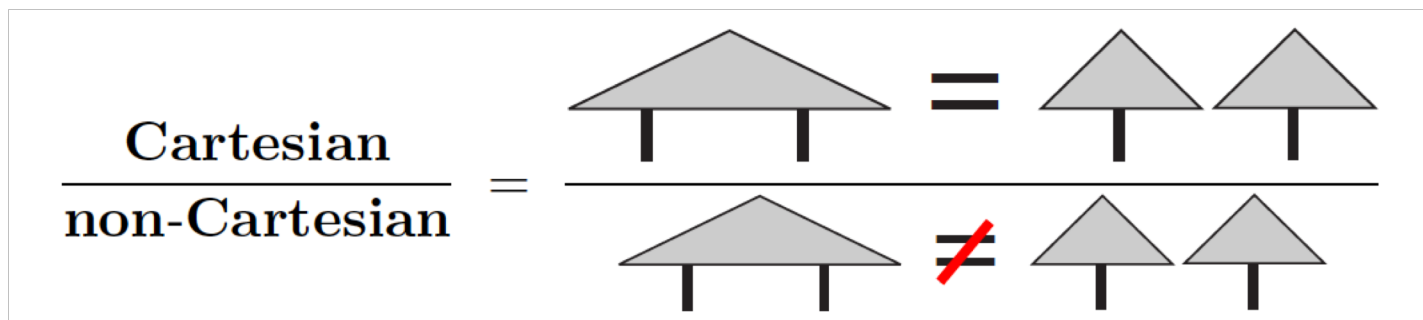

Graphical language for monoidal categories.

 $1_A \quad f$
 $g \circ f \quad 1_A \otimes 1_B$
 $f \otimes 1_C$
 $f \otimes g$
 $(f \otimes g) \circ h$

 $\psi : I \rightarrow A$
 $\pi : A \rightarrow I$
 $\pi \circ \psi : I \rightarrow I$

 $=$


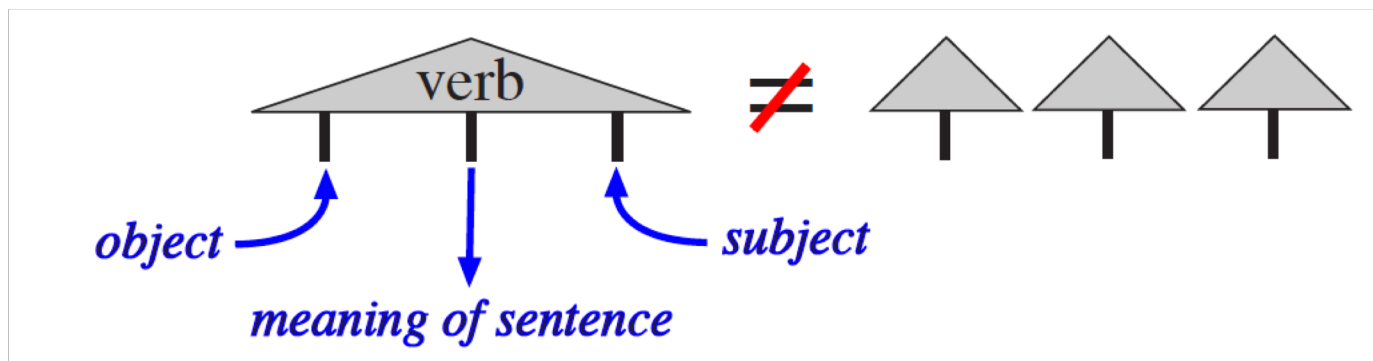
The 'from-meaning-of-words-to-meaning-of-a-sentence' process



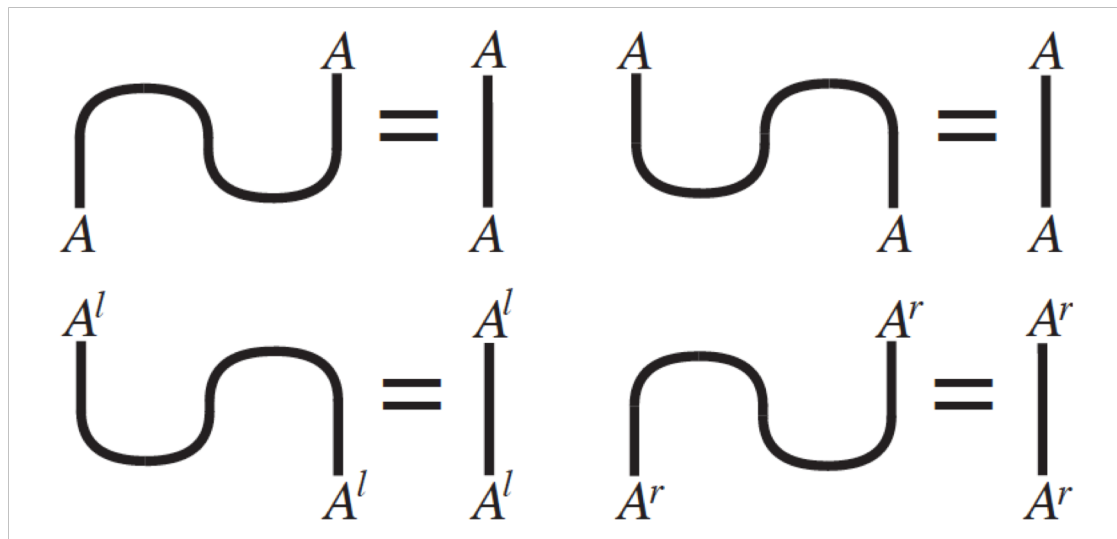
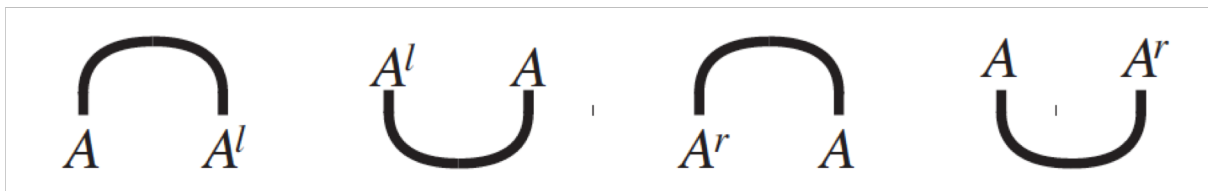
Compact closed categories



But if we consider a verb, then its grammatical type is $n^r s n^l$, that is, of the form $N \otimes S \otimes N$ within the realm of monoidal categories



Graphical language for compact closed categories.



Vector spaces, linear maps and tensor product as a compact closed category.

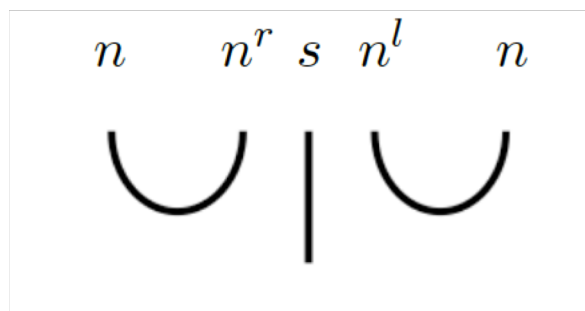
$$\eta^l = \eta^r : \mathbb{R} \rightarrow V \otimes V :: 1 \mapsto \sum_i \vec{e}_i \otimes \vec{e}_i$$

$$\epsilon^l = \epsilon^r : V \otimes V \rightarrow \mathbb{R} :: \sum_{ij} c_{ij} \vec{v}_i \otimes \vec{w}_j \mapsto \sum_{ij} c_{ij} \langle \vec{v}_i | \vec{w}_j \rangle .$$

$$\epsilon^l = \epsilon^r : V \otimes V \rightarrow \mathbb{R} :: \sum_{ij} c_{ij} \vec{e}_i \otimes \vec{e}_j \mapsto \sum_i c_{ii} .$$

A Pregroup as a compact closed category

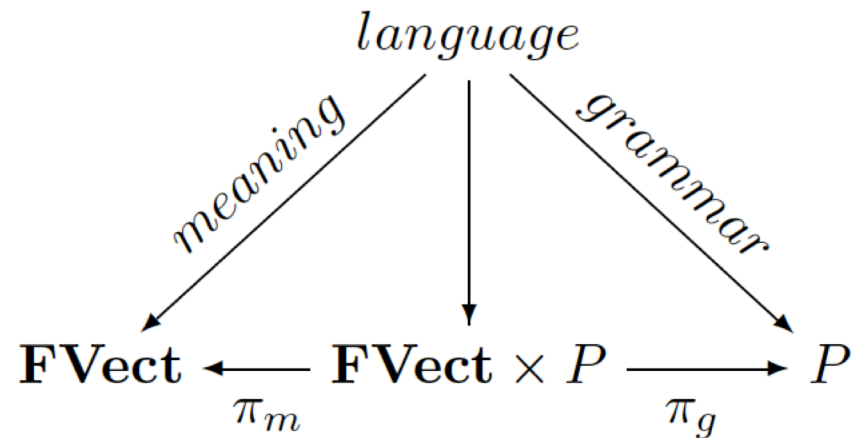
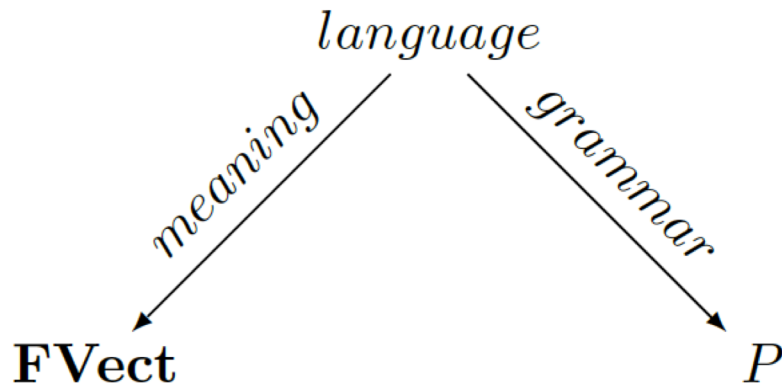
$$\begin{aligned}\eta^l &= [1 \leq p \cdot p^l] & \epsilon^l &= [p^l \cdot p \leq 1] \\ \eta^r &= [1 \leq p^r \cdot p] & \epsilon^r &= [p \cdot p^r \leq 1]\end{aligned}$$



$$\epsilon_n^r \otimes 1_s \otimes \epsilon_n^l : n \otimes n^r \otimes s \otimes n^l \otimes n \rightarrow s.$$

3.4 Categories representing both grammar and meaning

1. vector spaces can be used to assign meanings to words in a language;
2. Pregroups can be used to assign grammatical structure to sentences.



$$(f : V \rightarrow W, p \leq q)$$

$$(f, \leq) : (V, p) \rightarrow (W, q)$$

$$(\eta^l, \leq) : (\mathbb{R}, 1) \rightarrow (V \otimes V, p \cdot p^l) \qquad (\eta^r, \leq) : (\mathbb{R}, 1) \rightarrow (V \otimes V, p^r \cdot p)$$

$$(\epsilon^l, \leq) : (V \otimes V, p^l \cdot p) \rightarrow (\mathbb{R}, 1) \qquad (\epsilon^r, \leq) : (V \otimes V, p \cdot p^r) \rightarrow (\mathbb{R}, 1)$$

3.5 Meaning of a sentence as a morphism in $\mathbf{FVect} \times \mathbf{P}$.

Definition 3.2. We define the vector $\overrightarrow{w_1 \cdots w_n}$ of the meaning of a string of words $w_1 \cdots w_n$ to be

$$\overrightarrow{w_1 \cdots w_n} := f(\overrightarrow{w_1} \otimes \cdots \otimes \overrightarrow{w_n})$$

where for (W_i, p_i) meaning space of the word w_i , the linear map f is built by substituting each p_i in $[p_1 \cdots p_n \leq x]$ with W_i .

Thus for $\alpha = [p_1 \cdots p_n \rightarrow x]$ a morphism in $\mathbf{P}$ and $f = \alpha[p_i \setminus W_i]$ a linear map in $\mathbf{FVect}$, the following is a morphism in $\mathbf{FVect} \times \mathbf{P}$:

$$(W_1 \otimes \cdots \otimes W_n, p_1 \cdots p_n) \xrightarrow{(f, \leq)} (X, x)$$

We call f the ‘from-meaning-of-words-to-meaning-of-a-sentence’ map.

- Assign a grammatical type p_i to each word w_i of the string, apply the axioms and rules of the Pregroup grammar to reduce these types to a simpler type $p_1, \dots, p_n \rightarrow x$. If the string of words is a sentence, then the reduced type x should be the basic grammatical type s of a sentence
- Assign a vector space to each word of the sentence based on its syntactic type assignment. For the purpose of this paper, we prefer to be flexible with the manner in which these vector spaces are built, e.g. the vector spaces of the words with basic types like noun may be atomic and built according to the usual rules of the distributional model; the vector spaces of the words with compound types like verbs are tensor spaces.

- Consider the vector of the meaning of each word in the spaces built above, take their tensor, and apply to it the diagram of the syntactic reduction of the string, according to the meaning spaces of each word. This will provide us with the meaning of the string.

4 Computing the meaning of example sentences

4.1 Positive Transitive Sentence

A positive sentence with a transitive verb has the Pregroup type $n(n^r s n^l)n$.

We assume that the meaning spaces of the subject and object of the sentence are atomic and are given as $(V; n)$ and $(W; n)$.

The meaning space of the verb is compound and is given as $(V \otimes S \otimes W; n^r s n^l)$.

The 'from-meaning-of-words-to-meaning-of-a-sentence' linear map f is the linear map which realizes the following structural morphism in $\mathbf{FVect} \times \mathbf{P}$:

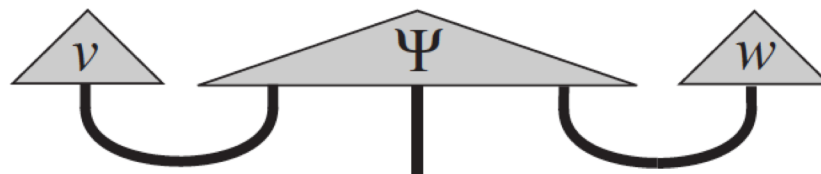
$$(V \otimes T \otimes W, n(n^r s n^l)n) \xrightarrow{(f, \leq)} (S, s)$$

$$f = \epsilon_V \otimes 1_S \otimes \epsilon_W : V \otimes (V \otimes S \otimes W) \otimes W \rightarrow S$$

$$V \otimes S \otimes W \cong V \otimes W \otimes S \cong V^* \otimes W^* \rightarrow S$$

$$V^* = V \text{ and } W^* = W$$

$$f(\vec{v} \otimes \vec{\Psi} \otimes \vec{w}) \in S \text{ for } \vec{v} \otimes \vec{\Psi} \otimes \vec{w} \in V \otimes S \otimes W$$



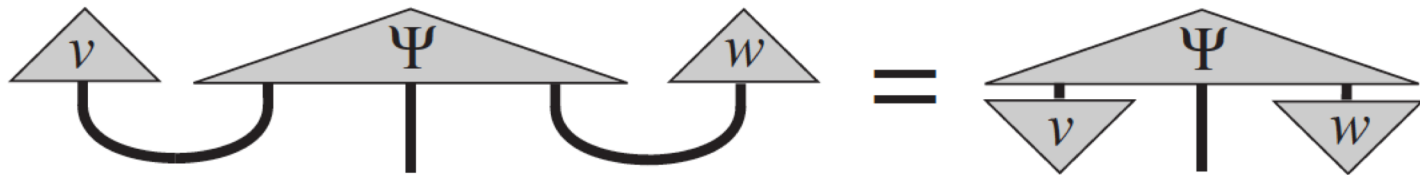
□ 一般に、動詞 Ψ について

$$\Psi = \sum_{ijk} c_{ijk} \vec{v}_i \otimes \vec{s}_j \otimes \vec{w}_k \in V \otimes S \otimes W$$

$$\begin{aligned} f(\vec{v} \otimes \vec{\Psi} \otimes \vec{w}) &= \epsilon_V \otimes 1_S \otimes \epsilon_W (\vec{v} \otimes \vec{\Psi} \otimes \vec{w}) \\ &= \sum_{ijk} c_{ijk} \langle \vec{v} | \vec{v}_i \rangle \vec{s}_j \langle \vec{w}_k | \vec{w} \rangle \\ &= \sum_j \left(\sum_{ik} c_{ijk} \langle \vec{v} | \vec{v}_i \rangle \langle \vec{w}_k | \vec{w} \rangle \right) \vec{s}_j \end{aligned}$$

Note that, in Dirac notation, $f(\vec{v} \otimes \vec{\Psi} \otimes \vec{w})$ is written as:

$$(\langle \epsilon_V^r | \otimes 1_S \otimes \langle \epsilon_V^r |) |\vec{v} \otimes \vec{\Psi} \otimes \vec{w}\rangle.$$



$$(\langle \vec{v} | \otimes 1_S \otimes \langle \vec{v} |) |\vec{\Psi}\rangle$$

Example 1. One Dimensional Truth-Theoretic Meaning. “John likes Mary”

$$\overrightarrow{John} \in V, \quad \overrightarrow{likes} \in T, \quad \overrightarrow{Mary} \in W$$

$$\overrightarrow{likes} = \sum_{ij} \overrightarrow{m}_i \otimes \overrightarrow{likes}_{ij} \otimes \overrightarrow{f}_j$$

$$\begin{aligned} f \left(\overrightarrow{m}_3 \otimes \overrightarrow{likes} \otimes \overrightarrow{f}_4 \right) &= \sum_{ij} \langle \overrightarrow{m}_3 | \overrightarrow{m}_i \rangle \overrightarrow{likes}_{ij} \langle \overrightarrow{f}_j | \overrightarrow{f}_4 \rangle \\ &= \sum_{ij} \delta_{3i} \overrightarrow{likes}_{ij} \delta_{j4} \\ &= \overrightarrow{likes}_{34} = \begin{cases} \overrightarrow{1} & m_3 \text{ likes } f_4 \\ \overrightarrow{0} & o.w. \end{cases} \end{aligned}$$

4.2 Negative Transitive Sentence "John does not like Mary"

John does not like Mary

$$n (n^r s j^l \sigma) (\sigma^r j j^l \sigma) (\sigma^r j n^l) n$$

$$f = (1_S \otimes \epsilon_J \otimes \epsilon_J) \circ (\epsilon_V \otimes 1_S \otimes 1_{J^*} \otimes \epsilon_V \otimes 1_J \otimes 1_{J^*} \otimes \epsilon_V \otimes 1_J \otimes \epsilon_W) : \\ V \otimes (V^* \otimes S \otimes J^* \otimes V) \otimes (V^* \otimes J \otimes J^* \otimes V) \otimes (V^* \otimes J \otimes W^*) \otimes W \rightarrow S$$

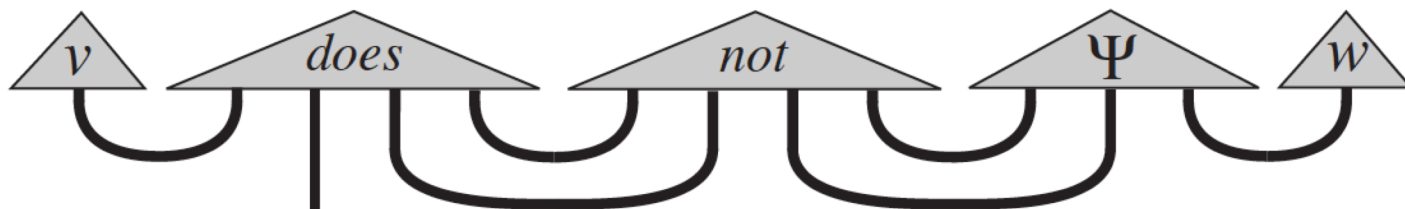
and depicted as:



When applied to the meaning vectors of words one obtains:

$$f(\vec{v} \otimes \overrightarrow{does} \otimes \overrightarrow{not} \otimes \vec{\Psi} \otimes \vec{w})$$

which is depicted as:

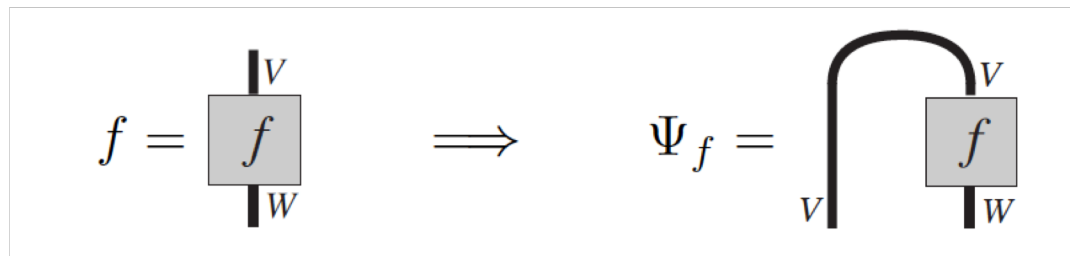


$$S = J \quad \text{and} \quad \overrightarrow{does} = \sum_{ii} \vec{e}_i \otimes \vec{e}_j \otimes \vec{e}_j \otimes \vec{e}_i \in V \otimes J \otimes J \otimes V.$$

$$\overrightarrow{does} \simeq (1_V \otimes \eta_J \otimes 1_V) \circ \eta_V : \mathbb{R} \rightarrow V \otimes J \otimes J \otimes V$$

There is a well-known bijective correspondence between linear maps of type $V \rightarrow W$ and vectors in $V \otimes W$. Given a linear map $f : V \rightarrow W$ then the corresponding vector is:

$$\Psi_f = \sum_i \vec{e}_i \otimes f(\vec{e}_i)$$



The trick to implement *not* will be to take this linear map to be the linear matrix representing the logical *not*. Concretely, while the matrix of the identity is $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, the matrix of the logical *not* is $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$. In Dirac notation, the vector corresponding to the identity is $|00\rangle + |11\rangle$, while the vector corresponding to the logical *not* is $|01\rangle + |10\rangle$. While we have

$$\overrightarrow{does} = \sum_i \vec{e}_i \otimes (|00\rangle + |11\rangle) \otimes \vec{e}_i \in V \otimes J \otimes J \otimes V,$$

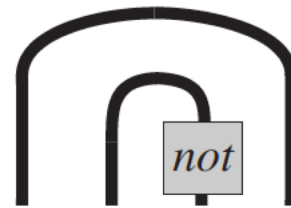
we will set

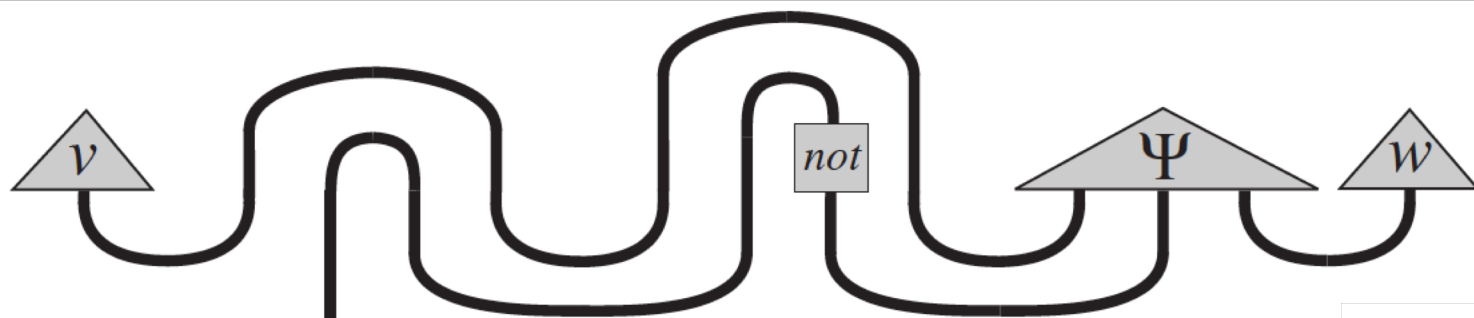
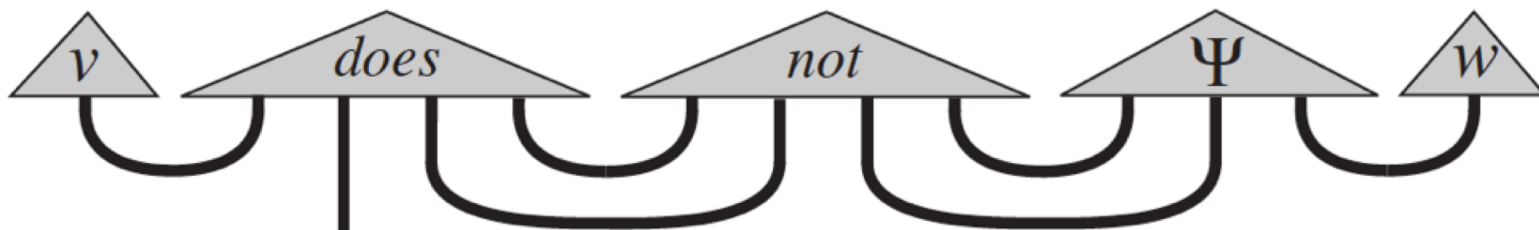
$$\overrightarrow{not} = \sum_i \vec{e}_i \otimes (|01\rangle + |10\rangle) \otimes \vec{e}_i \in V \otimes J \otimes J \otimes V.$$

$\overrightarrow{does} =$



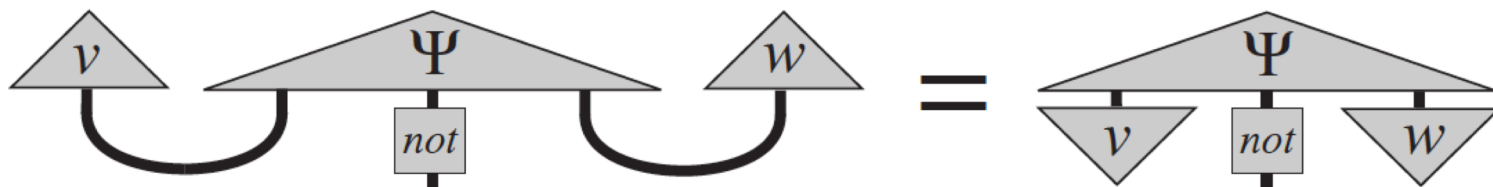
$\overrightarrow{not} =$





$$\left(\epsilon_V \otimes \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \otimes \epsilon_W \right) (\vec{v} \otimes \vec{\Psi} \otimes \vec{w})$$

$$\text{not} = \text{not}$$



Negative Truth-Theoretic Meaning

“John does not like Mary”

$$\overrightarrow{like} = \sum_{ij} \vec{m}_i \otimes \overrightarrow{like}_{ij} \otimes \vec{f}_j \quad \text{for} \quad \overrightarrow{like}_{ij} = \begin{cases} |1\rangle & m_i \text{ likes } f_j \\ |0\rangle & o.w. \end{cases}$$

Setting $N = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ we obtain:

$$(\epsilon_V \otimes N \otimes \epsilon_W) \left(\vec{m}_3 \otimes \overrightarrow{likes} \otimes \vec{f}_4 \right) = \sum_{ij} \langle \vec{m}_3 | \vec{m}_i \rangle N(\overrightarrow{likes}_{ij}) \langle \vec{f}_j | \vec{f}_4 \rangle =$$

$$\sum_{ij} \delta_{3i} N(\overrightarrow{likes}_{ij}) \delta_{j4} = N(\overrightarrow{likes}_{34}) =$$

$$\begin{cases} |1\rangle & \overrightarrow{like}_{34} = |0\rangle \\ |0\rangle & \overrightarrow{like}_{34} = |1\rangle \end{cases} = \begin{cases} |1\rangle & m_3 \text{ does not like } f_4 \\ |0\rangle & o.w. \end{cases}$$

5 Comparing meanings of sentences

□ Hierarchical Meaning.

$$\overrightarrow{likes} = \frac{3}{4}\overrightarrow{loves} + \frac{1}{4}\overrightarrow{hates}$$

□ Negative Hierarchical Meaning

□ Degree of similarity of positive sentences

$$\langle \overrightarrow{John\ loves\ Mary} \mid \overrightarrow{John\ likes\ Mary} \rangle = \frac{3}{4} \mid \overrightarrow{loves}_{34} \mid^2$$

$$\langle \overrightarrow{John\ hates\ Mary} \mid \overrightarrow{John\ likes\ Mary} \rangle =$$

$$\langle f(\vec{m}_3 \otimes \overrightarrow{hates} \otimes \vec{f}_4) \mid f(\vec{m}_3 \otimes \overrightarrow{likes} \otimes \vec{f}_4) \rangle = \frac{1}{4}$$

$$\langle \overrightarrow{John\ loves\ Mary} \mid \overrightarrow{John\ hates\ Mary} \rangle =$$

$$\langle f(\vec{m}_3 \otimes \overrightarrow{loves} \otimes \vec{f}_4) \mid f(\vec{m}_3 \otimes \overrightarrow{hates} \otimes \vec{f}_4) \rangle = 0.$$

□ Degree of similarity of negative sentences.

$$\begin{aligned}
 & \left\langle \overrightarrow{\text{John does not love Mary}} \mid \overrightarrow{\text{John does not like Mary}} \right\rangle = \\
 & \left\langle f(\vec{m}_3 \otimes \overrightarrow{\text{does}} \otimes \overrightarrow{\text{not}} \otimes \overrightarrow{\text{love}} \otimes \vec{f}_4) \mid f(\vec{m}_3 \otimes \overrightarrow{\text{does}} \otimes \overrightarrow{\text{not}} \otimes \overrightarrow{\text{like}} \otimes \vec{f}_4) \right\rangle \\
 & = \left\langle \overrightarrow{\text{hates}}_{34} \mid \frac{1}{4} \overrightarrow{\text{loves}}_{34} + \frac{3}{4} \overrightarrow{\text{hates}}_{34} \right\rangle \\
 & = \frac{1}{4} \left\langle \overrightarrow{\text{hates}}_{34} \mid \overrightarrow{\text{loves}}_{34} \right\rangle + \frac{3}{4} \left\langle \overrightarrow{\text{hates}}_{34} \mid \overrightarrow{\text{hates}}_{34} \right\rangle = \frac{3}{4}
 \end{aligned}$$

- Degree of similarity of positive and negative sentences.

$$\begin{aligned}
 & \langle f(\vec{m}_3 \otimes \overrightarrow{does} \otimes \overrightarrow{not} \otimes \overrightarrow{like} \otimes \vec{f}_4) \mid f(\vec{m}_3 \otimes \overrightarrow{likes} \otimes \vec{f}_4) \rangle \\
 &= \langle \frac{1}{4} \overrightarrow{loves}_{34} + \frac{3}{4} \overrightarrow{hates}_{34} \mid \frac{3}{4} \overrightarrow{loves}_{34} + \frac{1}{4} \overrightarrow{hates}_{34} \rangle \\
 &= (\frac{1}{4} \times \frac{3}{4}) \langle \overrightarrow{loves}_{34} \mid \overrightarrow{loves}_{34} \rangle + (\frac{3}{4} \times \frac{1}{4}) \langle \overrightarrow{hates}_{34} \mid \overrightarrow{hates}_{34} \rangle \\
 &= (\frac{1}{4} \times \frac{3}{4}) + (\frac{3}{4} \times \frac{1}{4}) = \frac{3}{8}
 \end{aligned}$$

Compositionality in Semantic Spaces

Martha Lewis

2019年

2nd Symposium on Compositional
Structures

[http://events.cs.bham.ac.uk/
syco/2/slides/lewis.pdf](http://events.cs.bham.ac.uk/syco/2/slides/lewis.pdf)



最新の話題

- Second Symposium on Compositional Structures (SYCO 2) University of Strathclyde
17-18 December, 2018 での発表。

- DisCo意味論の応用としての、「言い換え」「文の含意」。
これは、言語と論理の関係を考える時に重要。

『*My dog runs in the park*』 $\Leftarrow_{???}$ 『*My pet runs in the field*』

- ニューラルネットのアプローチを、DisCoから見直す。
この方向の延長上で、コネクショニズムを超えて、シンボルの形成を説明することができると僕は思う。
- 語の意味のベクトル表現を、positive operatorとして捉え返し、密度行列でconceptual space として表現する。
- 語のconceptual space表現の構築に、WordNet を使おう。

Applications (1/2)

Why are CDMs important?

The problem of producing robust representations for the meaning of phrases and sentences is at the heart of every task related to natural language.

- **Paraphrase detection**

Problem: Given two sentences, decide if they say the same thing in different words

Solution: Measure the cosine similarity between the sentence vectors

- **Sentiment analysis**

Problem: Extract the general sentiment from a sentence or a document

Solution: Train a classifier using sentence vectors as input

Applications (2/2)

- **Textual entailment**

Problem: Decide if one sentence logically infers a different one

Solution: Examine the feature inclusion properties of the sentence vectors

- **Machine translation**

Problem: Automatically translate one sentence into a different language

Solution: Encode the source sentence into a vector, then use this vector to decode a surface form into the target language

- *And so on. Many other potential applications exist...*

Quantizing the grammar

Coecke, Sadrzadeh and Clark (2010):

Pregroup grammars are structurally homomorphic with the category of finite-dimensional vector spaces and linear maps (both share **compact closure**)

- In abstract terms, there exists a structure-preserving passage from grammar to meaning:

$$\mathcal{F} : \text{Grammar} \rightarrow \text{Meaning}$$

- The meaning of a sentence $w_1 w_2 \dots w_n$ with grammatical derivation α is defined as:

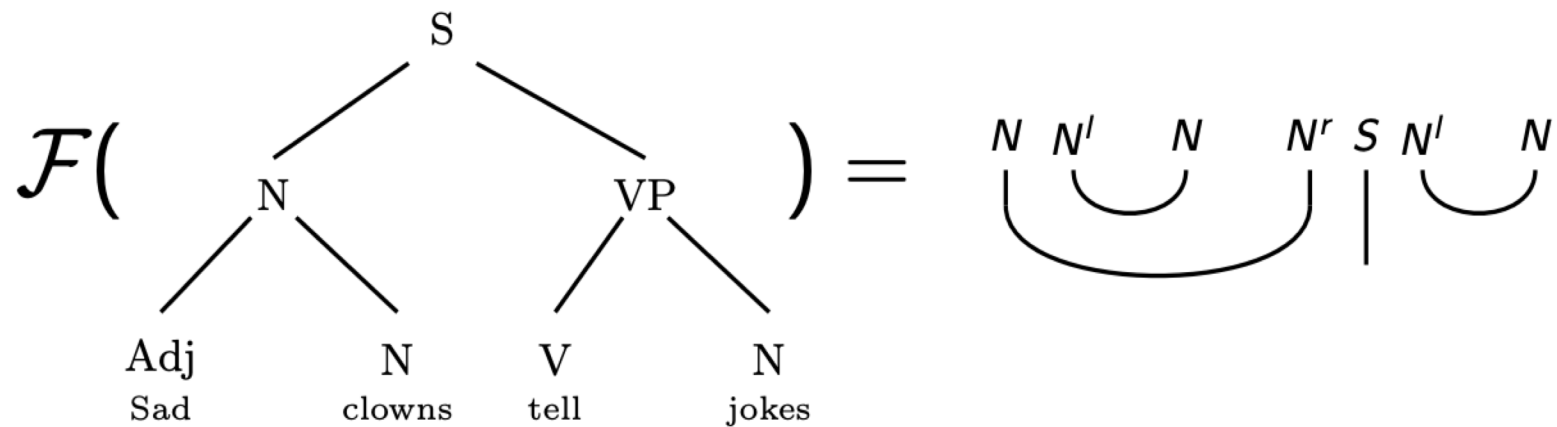
$$\overrightarrow{w_1 w_2 \dots w_n} := \mathcal{F}(\alpha)(\overrightarrow{w_1} \otimes \overrightarrow{w_2} \otimes \dots \otimes \overrightarrow{w_n})$$

A multi-linear model

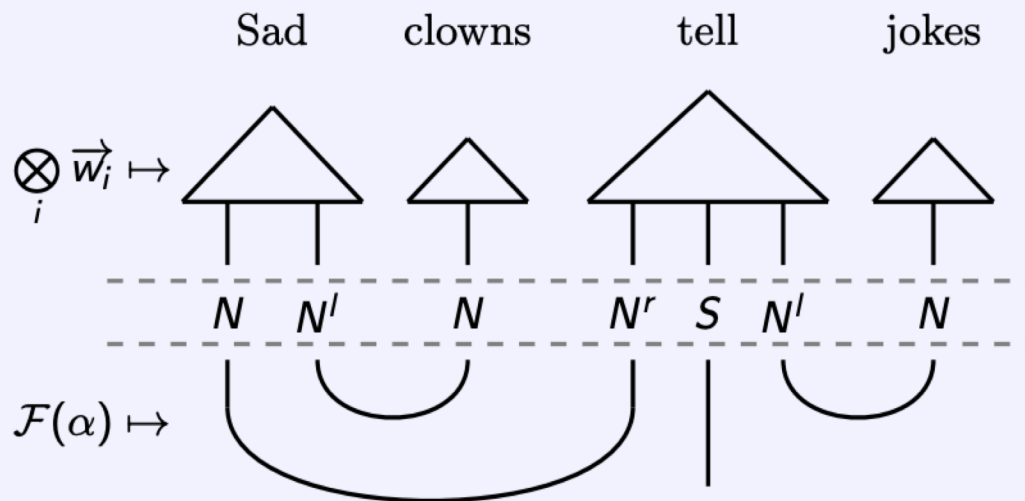
The grammatical type of a word defines the vector space in which the word lives:

- Nouns are vectors in N ;
 - adjectives are linear maps $N \rightarrow N$, i.e. elements in $N \otimes N$;
 - intransitive verbs are linear maps $N \rightarrow S$, i.e. elements in $N \otimes S$;
 - transitive verbs are bi-linear maps $N \otimes N \rightarrow S$, i.e. elements of $N \otimes S \otimes N$;
-
- The composition operation is **tensor contraction**, i.e. elimination of matching dimensions by application of inner product.

Graphical calculus: example



$\mathcal{F}(\alpha)(\overrightarrow{\text{trembling}} \otimes \overrightarrow{\text{shadows}} \otimes \overrightarrow{\text{play}} \otimes \overrightarrow{\text{hide-and-seek}})$



k -hyponymy interacts well with compositionality

- We would like our notion of entailment to work at the sentence level.
- Since sentences are represented as positive operators, we can compare them directly.
- If sentences have similar structure, we can also give a lower bound on the entailment strength between sentences based on the entailment strengths between the words in the sentences.

Example

Suppose $\llbracket \textit{dog} \rrbracket \preceq_k \llbracket \textit{pet} \rrbracket$ and $\llbracket \textit{park} \rrbracket \preceq_l \llbracket \textit{field} \rrbracket$. Then

$$\llbracket \textit{My dog runs in the park} \rrbracket \preceq_{???} \llbracket \textit{My pet runs in the field} \rrbracket$$

How should we build density matrices for words?

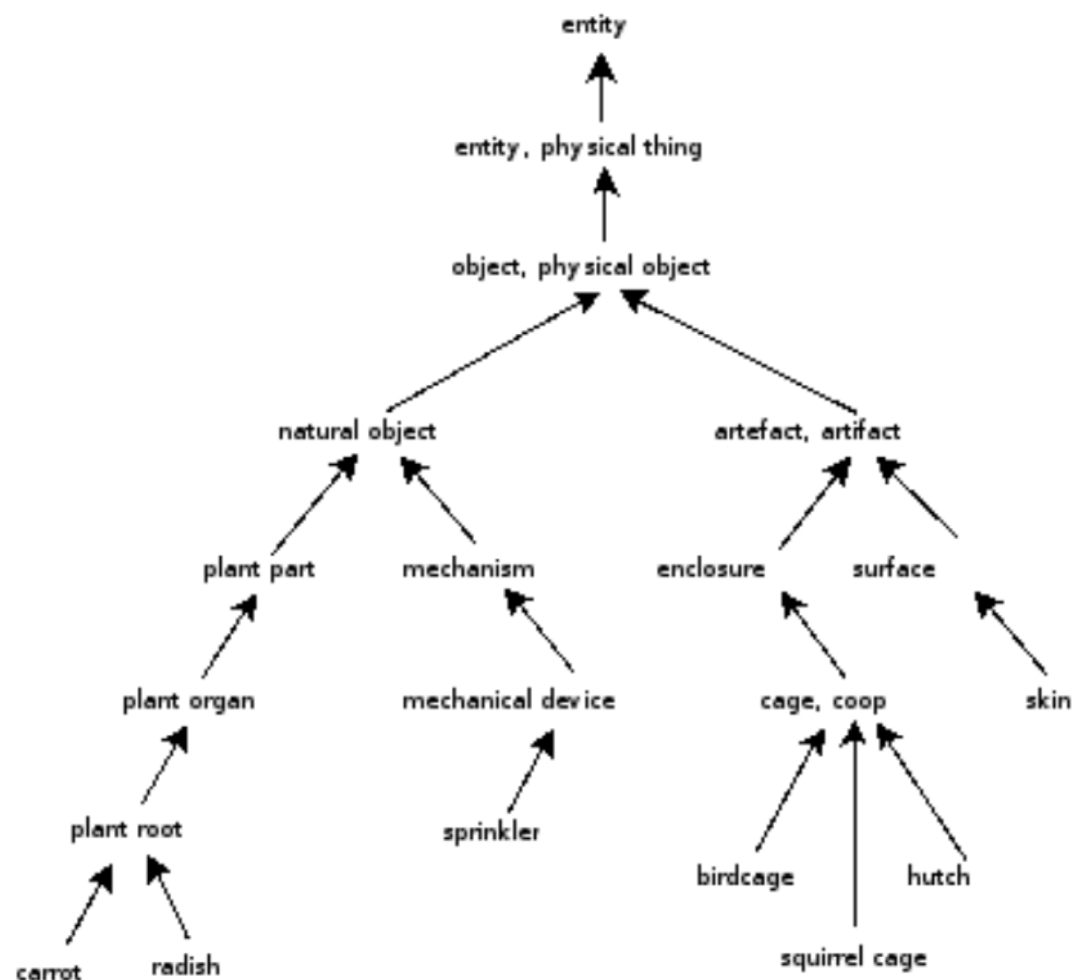


Figure 1. "is a" relation example

WordNet. Princeton University. 2010

- Understanding neural networks as a semantics category for Lambek categorial grammar opens up more possibilities to use tools from formal semantics in computational linguistics.
- We can immediately see possibilities for building alternative networks.
- Decomposing the tensors for functional words into repeated applications of a compositionality function gives options for learning representations.
- Brittleness of word types in DisCo is alleviated

Summary

- Vector spaces provide a good means of talking about similarity between words.
- But both conceptual spaces and positive operators give us a richer word representation.
- We don't have a good way of building word representations in conceptual spaces yet, but using information from WordNet to build density matrices seems to give good results with very little effort.

おわりに

- DisCO意味論の達成は、まだまだプリミティブなものだ。それは、むしろ、現在の言語の理論そのものが、発展史的には未完成であることを示していることの反映である。
- ただ、文法のカテゴリーから意味のカテゴリーへのFunctorとして意味のモデルを与えるというアプローチは、大きな可能性を持っていると、僕は考えている。
- 最後に紹介したMartha Lewisの講演は、DisCOに対する僕の不満や関心に多くの点で応えるものだった。
 - 言語能力と論理・数学的認識能力の関係
 - コネクショニズムと計算主義をつなぐ道
 - 「概念のリソース」としての認識能力と言語能力
- こうした、いわば認識論的課題に応えることが、期待される。

おわりに

- DisCO意味論は、Appendixに見るように、量子情報理論への接近を進めているようにも見えるが、それが深いものであるかは、僕は判断を留保している。
- 認識と意味の理論として、カテゴリー論を援用しようとするのなら、LawvereやGrothendieckやVoevodskyの認識的関心に遡及していくのが実りの多い道だと僕は考えている。

1月29日「楽しい数学」第四回

「同じ」を考える – 型の理論入門

楽しい数学 第四夜



Voevodsky 追悼



「同じ」を考える
-- 型の理論入門 --

Appendix

Quantum Semantics



Appendix Quantum Semantics

- A quantum teleportation inspired algorithm produces sentence meaning from word meaning and grammatical structure
 - Open System Categorical Quantum Semantics in Natural Language Processing
-

A quantum teleportation inspired
algorithm produces sentence meaning
from word meaning and grammatical
structure

Stephen Clark, Bob Coecke, Edward Grefenstette,
Stephen Pulman, Mehrnoosh Sadrzadeh

2013年

<https://arxiv.org/pdf/1305.0556.pdf>

Abstract

We discuss an algorithm which produces the meaning of a sentence given meanings of its words, and its resemblance to quantum teleportation. In fact, this protocol was the main source of inspiration for this algorithm which has many applications in the area of Natural Language Processing.

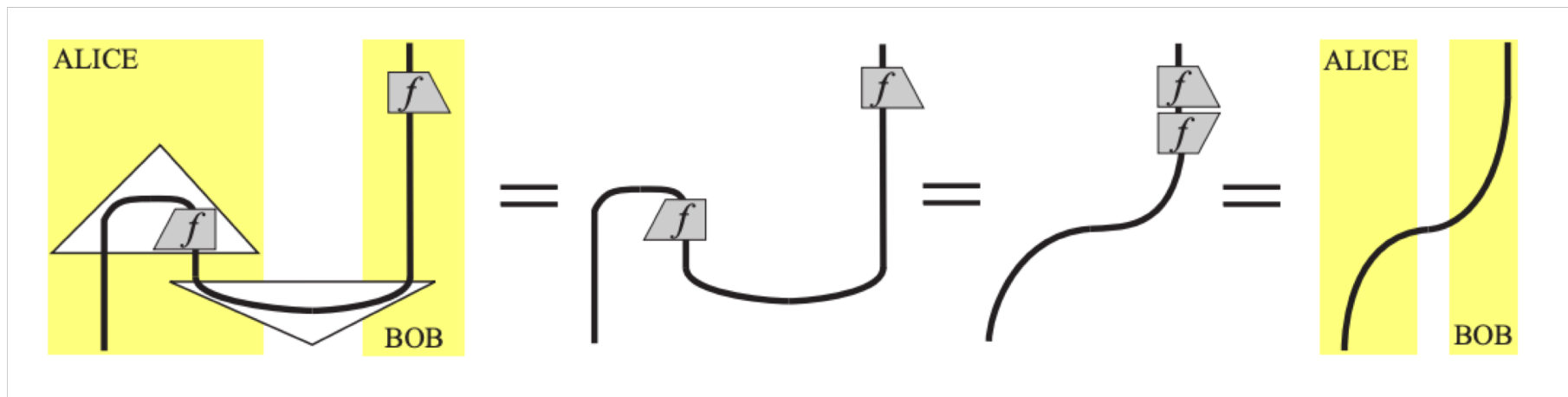
The NLP problem

- Dictionaries explain the meanings of words; however, in natural language words are organized as sentences, but we don't have dictionaries that explain the meanings of sentences. Still, a sentence carries more information than the words it is made up from; e.g. $\text{meaning}(\text{Alice sends a message to Bob}) \neq \text{meaning}(\text{Bob sends a message to Alice})$.

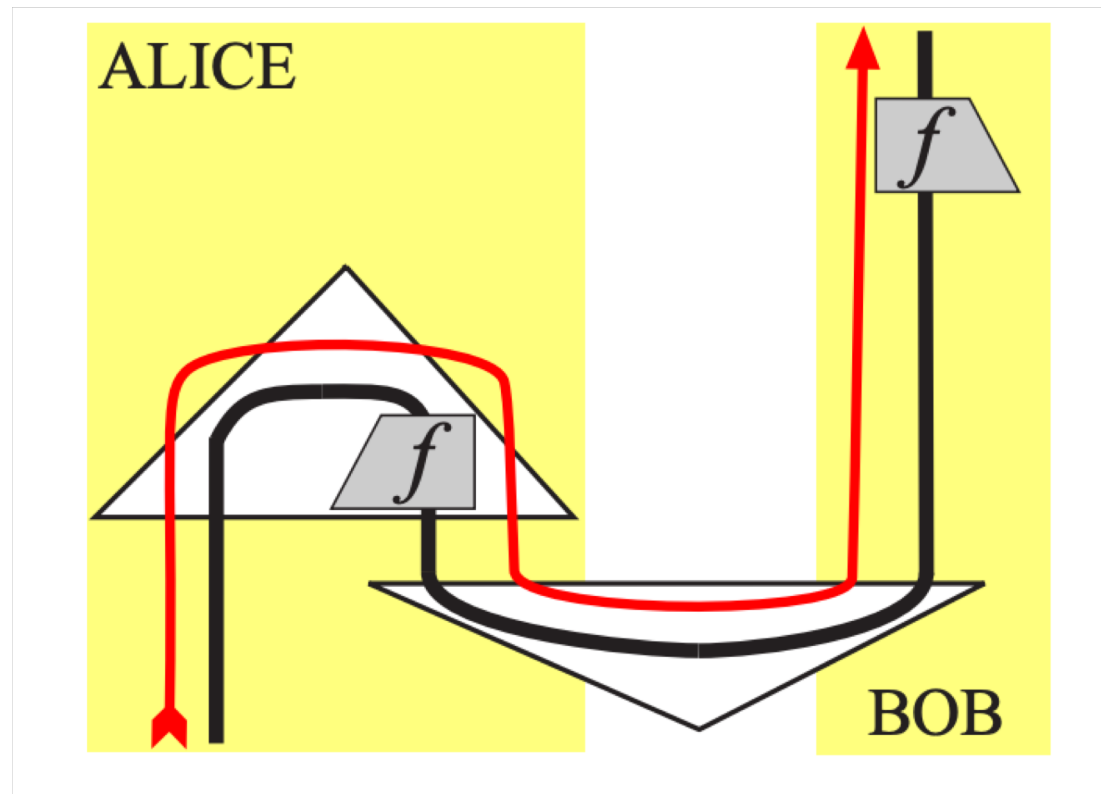
two separate communities

- Until recently, the state of affairs in computational linguistics was one of two separate communities [13]. One community focused on non-compositional purely distributional methods such as the vector space model described above. The other community studied the compositional mathematical structure of sentences, building on work by Chomsky [5], Lambek [18] and Montague [21]. This work is mainly about the grammatical structure of sentences; grammatical type calculi are algebraic gadgets that allow one to verify whether a sentence has a correct grammatical structure.

Caps, cups, and teleportation



- Here, the red line represents the logical flow which indicates that the state incoming at Alice's side first gets acted upon by an operation f , and then by its adjoint $f^\dagger$, which in the case that f is unitary results in the outgoing state at Bob's side being identical to the incoming one at Alice's side.



When interpreted in Hilbert space, the key ingredients of this formalism are ‘cups’ and ‘caps’:

$$\text{cup} := |00\rangle + |11\rangle \quad \text{cap} := \langle 00| + \langle 11|$$

and the equation that governs them is:

$$((\langle 00| + \langle 11|) \otimes Id)(Id \otimes (|00\rangle + |11\rangle)) = Id$$

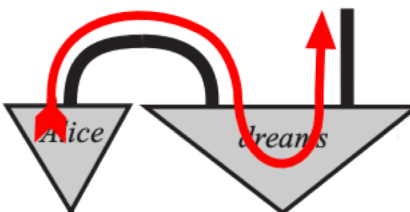
which diagrammatically depicts as:

The diagram shows a cap (a thick black arc opening upwards) followed by a cup (a thick black arc opening downwards). These two arcs are enclosed within a dashed rectangular box. To the right of this box is an equals sign, followed by another dashed rectangular box containing a single vertical line, representing the identity operator.

Solution to the NLP problem: the intuition

$$\overrightarrow{Alice} \otimes \overrightarrow{hates} \otimes \overrightarrow{Bob} =$$




$$\left(\sum_i \langle ii| \otimes Id \right) (\overrightarrow{Alice} \otimes \overrightarrow{dreams}) =$$


$$\begin{array}{l}
 \text{e.g. Alice} \quad \text{e.g. hates} \quad \text{e.g. Bob} \\
 \underbrace{n} \cdot \underbrace{{}^{-1}n \cdot s \cdot n^{-1}} \cdot \underbrace{n} = (n \cdot {}^{-1}n) \cdot s \cdot (n^{-1} \cdot n) \\
 \leq 1 \cdot s \cdot 1 \\
 = s
 \end{array}$$

$n \cdot {}^{-1}n \leq 1$ and $n^{-1} \cdot n \leq 1$ can also be represented with ‘directed’ caps:

$$\begin{array}{c}
 s \\
 \curvearrowright \quad | \quad \curvearrowleft \\
 n \cdot {}^{-1}n \cdot s \cdot n^{-1} \cdot n
 \end{array}$$

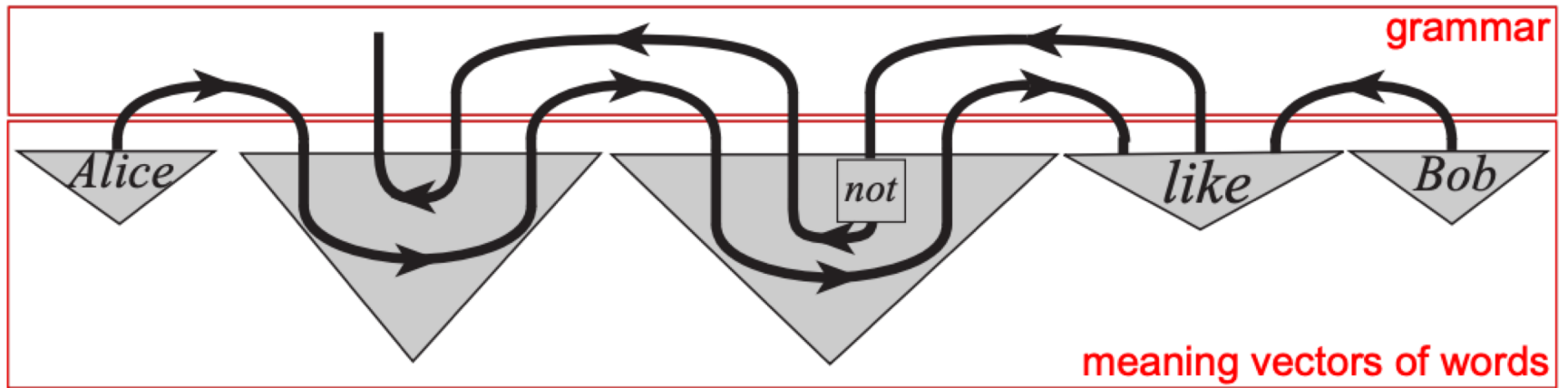
which represent the inequalities:

$$\begin{array}{ccc}
 \curvearrowright & \Leftrightarrow & \begin{array}{c} 1 \\ \vee | \\ n \cdot {}^{-1}n \end{array} & \quad & \curvearrowleft & \Leftrightarrow & \begin{array}{c} 1 \\ \vee | \\ n^{-1} \cdot n \end{array}
 \end{array}$$

Solution to the NLP problem: the algorithm.

$$n \rightsquigarrow \mathcal{V} \qquad tv = {}^{-1}n \cdot s \cdot n^{-1} \rightsquigarrow \mathcal{V} \otimes \mathcal{W} \otimes \mathcal{V}$$

1. Compute the tensor product $\overrightarrow{Words} = \overrightarrow{v}_1 \otimes \dots \otimes \overrightarrow{v}_k$ of the word meaning vectors in order of appearance in the sentence; e.g. $\overrightarrow{noun}_1 \otimes \overrightarrow{verb} \otimes \overrightarrow{noun}_2$.
2. Construct a linear map f that represents the type reduction as follows: given the diagram that represents a type reduction (cf. (1) above), we interpret caps as $\sum_i \langle ii |$ and straight wire as identities; e.g. $\sum_i \langle ii | \otimes Id \otimes \sum_i \langle ii |$.
3. Compute $\overrightarrow{Sentence} := f(\overrightarrow{Words}) \in \mathcal{W}$.



Open System Categorical Quantum Semantics in Natural Language Processing

Robin Piedeleu, Dimitri Kartsaklis, Bob Coecke and
Mehrnoosh Sadrzadeh

<https://arxiv.org/pdf/1502.00831v2.pdf>