# Aggregate Noun Phrases - Exercises

Camilo Thorne

ESSLLI 2013, Düsseldorf Aug 5-9, 2013

This sheet contains some simple exercises regarding the notions covered in the course, plus a couple on Turing machines and computational complexity. Some exercises will be discussed during the next lesson, if time permits. The remaining ones will be discussed during the lab session (i.e., Friday).

## Day One – Formal Semantics

**Quantifier characterizations.** Several definitions of generalized quantifiers exist in the literature. We consider only quantifiers of type (1,1). Let $\Psi_Q$ denote the "characterizing condition" of quantifier $Q$.

**(1)** Given a (finite) model $\mathcal{I}$, a generalized quantifier $Q$ of type (1,1) is the relation over $A, B$ where $A, B \subseteq \Delta$ verify condition $\Psi_Q$.

**(2)** A generalized quantifier $Q$ is the class of all (finite) models $\mathcal{I}$ closed under isomorphisms where $A, B \subseteq \Delta$ verify condition $\Psi_Q$.

**(3)** Given a (finite) model $\mathcal{I}$, a generalized quantifier $Q$ is a higher-order Boolean-valued function over $\mathcal{P}(\Delta) \times \mathcal{P}(\Delta)$ that takes two sets $A, B \subseteq \Delta$ and returns "true" iff $A$ and $B$ verify condition $\Psi_Q$.

Show that **(1)** – **(3)** are equivalent.

**Quantifier monotonicity.** Given $\mathcal{I}$, a generalized quantifier $Q$ over $A, B$ can be seen as a two-argument function ($A$ being the first, $B$, the second). We say that $Q$ is *monotonic* in its first argument iff given $A, B, C \subseteq \Delta$, either

- if $(A, B) \in Q$ and $A \subseteq C$, then $(C, B) \in Q$ (upward monotonic);
- if $(A, B) \in Q$ and $C \subseteq A$, then $(C, B) \in Q$ (downward monotonic).

We say that it is monotonic in its second argument if the previous condition holds also for $B$.

1. Show that the generalized quantifier "every" is downward monotonic in its first argument, and that "some" is upward monotonic in both arguments.
2. Is "most", of semantics $[\![\text{most}]\!] = \{(A, B) \subseteq \Delta \times \Delta \mid \#(A \cap B) \geq cardA \setminus B\}$ upward monotonic, downward monotonic or neither in its first argument?

## Day Two – Semantics of Aggregation

**Quantifier monotonicity (ctd.).** Determine if aggregate quantifiers are monotonic in either of their arguments, viz.,

1. If "the $\beta$-est" is upward or downward monotonic (or neither) in either of their arguments;
2. If "the total $\beta$ of ", "the product $\beta$ of", "the number of", "the average $\beta$ of " are upward or downward monotonic (or neither) in either of their arguments.

**Monotonicity of aggregation.** We say that an aggregate function $\alpha(\cdot)$ is upward or downward *monotonic* iff for all $A, B \subseteq \Delta$, if $A \subseteq B$, then $\alpha(A) \leq \alpha(B)$.

1. Determine if **argmin**$(\cdot)$, **argmax**$(\cdot)$, **count**$(\cdot)$, **avg**$(\cdot)$ and **prod**$(\cdot)$ are upward or downward
2. What happens when $A$ and $B$ are *bags* over $\Delta$?
3. What happens if we consider different number domains, viz., $\mathbb{R}$, $\mathbb{N}$, $\mathbb{Z}$ or $\mathbb{Q}$?

## Day Three – Semantic Complexity

**Turing Machines.** Define a Turing machine $M$ with tape alphabet $\Gamma = \{\star, L, R\}$ that decides the problem $\mathbf{CF} = \{0^i 1^i \mid i \geq 0\}$. In so doing:

1. Estimate $t(M)$ and $s(M)$ (use big $O$ notation if possible).
2. What is the theoretical complexity optimum (hint: remember Chomsky).

**NP-completeness.** A Problem $\mathcal{P}$ can be *polynomially reduced* to a problem $\mathcal{P}'$, in symbols $\mathcal{P} \leq_p \mathcal{P}'$, iff there exists an algorithm $\textsc{Alg}(\cdot)$ such that, for all $w \in \{0,1\}^*$, (i) $w \in \mathbf{P}$ iff $\textsc{Alg}(w) \in \mathbf{P}'$, (ii) $\textsc{Alg}(\cdot)$ runs in time $O(p(|w|)$, where $p(\cdot)$ denotes a polynomial of some degree $k$.

Modulo polynomial reductions, the the class of NP-complete problems can be seen as the (equivalence) class of the mutually polynomially reducible problems in NP, of which the problem **3-SAT**, defined as follows, is the representative: **(1) Instance:** A conjunction $\varphi$ of $k$ clauses (propositional disjunctions) of the form

$$\bigwedge_{i=0}^{k} \pm p_{i,1} \vee \pm p_{i,2} \vee \pm p_{i,3},$$

where $\pm p_{i,j}$ is a positive or negative propositional atom (a *literal*). **(2) Question:** Is $\varphi$ satisfiable?

An independent set in a graph $G = (V, E)$ is a subset $V' \subseteq V$ of disconnected vertexes in $G$, i.e., s.t. for all $u, v \in V'$, $(u.v) \notin E$. The **k-IND** problem is defined as follows: **(1) Instance:** A graph $G = (V, E)$ and $k \leq \#(V)$. **(2) Question:** Does there exist an independent set $V'$ of size $\#(V') \geq k$ in $G$?

1. Show that **k-IND** is NP-hard, viz., show that **k-IND** $\leq_p$ **3-SAT**.
2. Show that **k-IND** $\in$ NP. What do you conclude?

**Quantifiers.** We recall that the semantic complexity of the quantifier consists in the data complexity of its associated model checking problem.

1. Prove that the quantifier "most" is expressible in terms of aggregate quantifiers and hence has LSpace semantic complexity.
2. Prove that when we combine the quantifiers "more than 1/3" and "each other" (i.e., we "Ramseyfy" the former), semantic complexity becomes NP-hard.

## Day Four – Distributions

Repeat the experiment seen during the 4th session as follows. Using the scripting language of your choice, define regular expressions that recognize instances of

- the quantifier "exactly one" and
- the quantifier "most";

over the Brown corpus, and compute thereafter their relative frequency. A bar plot is desirable, but not necessary. Raw frequency will do too.

How does frequency correlate to their respective semantic complexity? Note that the quantifier "exactly one", called sometimes in the literature the *descriptor* operator (as it introduces so-called "definite descriptions"), can be expressed by several surface forms, such as "exactly one", "the" (as in "the queen of England"), "a unique", "the unique", etc. You will also need to determine the theoretical semantic complexity of "exactly one".