nanobird

The nanobird Relevancy Engine

A Context Analyzer and Suggestion System

Ihab Hamadeh and Zach Hajjaj

Table of Contents

List of Figures

1 The nanobird Relevancy Engine

The *nanobird Relevancy Engine* (nRE) is a system that analyzes an inputted natural language data (text) and extracts a set of phrases that describes and/or summarizes the text. The nRE generates a graph showing the relations between phrases and deduces the context in text. The *nanobird Relevancy Engine* (nRE) is composed of three parts: a set of phrase dictionaries, a phrase tokenizer, and a phrase relevancy/context analyzer (Figure 1).

1.1 Definitions

Word. A single distinct meaningful element of speech or writing, used with others (or sometimes alone) to form a sentence and typically shown with a space on either side when written or printed.

Phrase. A single word or a group of words.

Alias phrases. Phrases that take many different forms and still refer to the same thing/concept.

Ambiguous phrase. A phrase that could refer to many different thing/concept.

1.2 nanobird Phrase Dictionaries

To create a context-aware natural language analyzer, different sets of phrase dictionaries are built, each acting as a cornerstone to the *nanobird* Relevancy Engine. We define/build five different dictionaries: a dictionary of unambiguous phrases (D_U), a dictionary of ambiguous phrases (D_A), a dictionary of phrase aliases (D_L), a dictionary of stop phrases (D_S) and a dictionary of phrase fingerprints (D_F).

Each dictionary is an abstract data type composed of a collection of (key,value) pairs such that each possible key appears at most once in the collection. Two different lookups can be performed:

- a lookup to check if a given key exists in the dictionary
- a lookup to find the value (if any) that is bound to a given key

It is important to note that the sets of keys $K_{D_U}, K_{D_A}, K_{D_L}$ of the dictionaries D_U , D_A , and D_L respectively are unique across these dictionaries, that is:

$$K_{D_U} \cap K_{D_A} \cap K_{D_L} = \varnothing$$

1.2.1 Dictionary of unambiguous phrases (D_U)

Generally, a relevancy system requires a dictionary of predefined phrases obtained from different sources such as encyclopedias and glossaries. The unambiguous phrases dictionary (D_U) consists of predefined phrases extracted from Wikipedia, a free, web-based, collaborative, multilingual encyclopedia.

Each phrase refers to a thing/person/concept that is not open to more than one interpretation. An example of such phrases is *“Billy Apple”, “Global warming”, and “Apple Inc.”* which refer to something/someone specific. Figure 2 shows a subset of phrases found in the unambiguous phrases dictionary.

Each key (k_{D_U}) in the dictionary D_U consists of a phrase such as “Global warming” and “Apple Inc.”. The value associated with each key is not essential to the current implementation.

1.2.2 Dictionary of ambiguous phrases (D_A)

In natural languages, some phrases are open to more than one interpretation and thus may refer to different things/concepts. Learning the different meanings of a phrase is essential to the understanding of a natural language. For instance, the word *“Apple”* can mean any of these following concepts:

- *Apple Inc.*
- *Apple (fruit)*
- *Fiona Apple*
- *Apple Store*

The dictionary of ambiguous phrases provides a list of ambiguous phrases along with the different possible interpretations. Each key k_{D_A} in the dictionary D_A consists of a phrase such as “Apple”. The value associated with each key is a list of unambiguous phrases in D_U (Figure 3).

$$\{k, v\}_{D_A} = \{Apple, [Apple\ Inc., Apple\ (fruit), Fiona\ Apple, Apple\ Store, \ldots]\}$$

1.2.3 Dictionary of phrase aliases (D_L)

Some phrases may be presented in different forms and still refer to the same thing/concept. These forms range from a simple few character change to a whole phrase change. A dictionary of phrase aliases (D_L) is a mapping of each phrase (k_{D_L}) to an unambiguous phrase in D_U . For instance, *“Global wobble”* is an alias to the phrase *“Global warming”*. Figure 4 gives different examples of phrase aliases and their mapping to their equivalent unambiguous phrases.

$$\{k_i, v\}_{D_L} = \{Climate\ crisis, Global\ warming\}$$

$$\{k_j, v\}_{D_L} = \{Planetary\ warming, Global\ warming\}$$

$$\{k_k, v\}_{D_L} = \{Stats, Statistics\}$$

1.2.4 Dictionary of stop phrases (D_S)

The dictionary of stop phrases (D_S) consists of phrases that are filtered out prior to or during the processing of natural language data (text). It is controlled by human input and not automated. Any group of words can be selected as the stop phraes to avoid incorrect context analysis. Examples of stop phrases are "*the*", "*is*", "*Wednesday*", "*which*", and "*lots of*".

Each key (k_{D_S}) in the dictionary D_S consists of phrases such as "*the*", "*is*", "*Wednesday*", "*which*", and "*lots of*". The value associated with each key is not essential to the current implementation.

1.2.5 Dictionary of phrase fingerprints (D_F)

A phrase fingerprint is a map representing a phrase and its distance to other relevant phrases.

Figure 5 is an example showing the fingerprint of the phrase "Apple Inc.".

Each key k_{D_F} in the dictionary of phrase fingerprints (D_F) consists of a pair of phrases found in unambiguous phrase dictionary (D_U). The value associated with each key represents the distance between the pair of phrases.

$$\{k_i, v\}_{D_F} = \{(Apple\ Inc., iPad),\ 0\}$$

$$\{k_j, v\}_{D_F} = \{(Apple\ Inc., Mac\ OS\ X),\ 1\}$$

$$\{k_k, v\}_{D_F} = \{(Apple\ Inc., Personal\ computer),\ 3\}$$

$$\{k_l, v\}_{D_F} = \{(Apple\ Inc., Mac\ OS\ X\ Lion),\ -6\}$$

To build a dictionary of phrase fingerprints (D_F), encyclopedia articles, namely Wikipedia articles, are processed using a proprietary algorithm that correlates phrases within an article and across articles.

1.2.5.1 Phrase Correlation Algorithm

The nanobird Phrase Correlation Algorithm is based on building relations between phrases in Wikipedia articles.

A Wikipedia article can be split into five sections (Figure 6):

- a definition (first paragraph),
- an information box,
- a description (remaining paragraphs up to the first outline),

- a summary of information (the remaining paragraphs of the article), and
- categories section.

Each section contains a set of phrases some of which are relevant to the phrase the article is about. The sets of phrases are built by collecting the phrases in the hyperlink from each section. Once the phrases are collected, they are passed to the nanobird Phrase Correlation algorithm. The algorithm is based on a set of definitions described in the remaining of this section.

1.2.5.1.1 Notations

The following notations are used in describing the Phrase Correlation algorithm

- ω : the phrase (thing/person/concept) the article is about.
- A_{ω} : the article that defines/describes a thing, person or concept.
- $\eta^{A}_{(\omega)}, \eta^{B}_{(\omega)}, \eta^{C}_{(\omega)}, \eta^{D}_{(\omega)}, \eta^{E}_{(\omega)}$: the set of phrases and their aliases in the five sections of A_{ω} (Figure 6).
- $\eta_{(\omega)}$: the complete set of phrases for all sections of A_{ω} such that

$$\eta_{(\omega)} = \eta^{A}_{(\omega)} \cup \eta^{B}_{(\omega)} \cup \eta^{C}_{(\omega)} \cup \eta^{D}_{(\omega)} \cup \eta^{E}_{(\omega)}$$

- $\lambda^{A}_{i(\omega)}$: the i^{th} hyperlinked phrase in the "definition" $\eta^{A}_{(\omega)}$
- D : the dictionary of predefined phrases which consists of the combination of the three dictionaries D_U, D_A, D_L

$$D = D_U \cup D_A \cup D_L$$

- $\Lambda^{A}_{(\omega)}$, $\Lambda^{B}_{(\omega)}$, $\Lambda^{C}_{(\omega)}$, $\Lambda^{D}_{(\omega)}$: the set of hyperlink phrases in $\eta^{A}_{(\omega)}$ $\eta^{B}_{(\omega)}$, $\eta^{C}_{(\omega)}$, and $\eta^{D}_{(\omega)}$ respectively such that:

$$\Lambda^{A}_{(\omega)} = \left\{\lambda^{A}_{1(\omega)}, \lambda^{A}_{2(\omega)}, \cdots, \lambda^{A}_{(n-1)(\omega)}, \lambda^{A}_{n(\omega)}\right\} \text{ and } \Lambda^{A}_{(\omega)} \subseteq \eta^{A}_{(\omega)}$$

$$\Lambda^{B}_{(\omega)} = \left\{\lambda^{B}_{1(\omega)}, \lambda^{B}_{2(\omega)}, \cdots, \lambda^{B}_{(n-1)(\omega)}, \lambda^{B}_{n(\omega)}\right\} \text{ and } \Lambda^{B}_{(\omega)} \subseteq \eta^{B}_{(\omega)}$$

$$\Lambda^{C}_{(\omega)} = \left\{\lambda^{C}_{1(\omega)}, \lambda^{C}_{2(\omega)}, \cdots, \lambda^{C}_{(n-1)(\omega)}, \lambda^{C}_{n(\omega)}\right\} \text{ and } \Lambda^{C}_{(\omega)} \subseteq \eta^{C}_{(\omega)}$$

$$\Lambda^{D}_{(\omega)} = \left\{\lambda^{D}_{1(\omega)}, \lambda^{D}_{2(\omega)}, \cdots, \lambda^{D}_{(n-1)(\omega)}, \lambda^{D}_{n(\omega)}\right\} \text{ and } \Lambda^{D}_{(\omega)} \subseteq \eta^{D}_{(\omega)}$$

It is important to note that all hyperlinked phrases are part of the dictionary of predefined phrases.

$$\omega \in D \ , \ \Lambda^{A}_{(\omega)} \subseteq D \ , \ \Lambda^{B}_{(\omega)} \subseteq D \ , \ \Lambda^{C}_{(\omega)} \subseteq D \ , \ \Lambda^{D}_{(\omega)} \subseteq D$$

- $\delta_{\omega_i \to \omega_j}$: a signed distance between two phrases ω_i and ω_j ; thus $\delta_{\omega_i \to \omega_j} = -\delta_{\omega_j \to \omega_i}$.

We define 15 different distances between two phrases.

$$0 \leq \left| \delta_{\omega_i \to \omega_j} \right| \leq 14$$

The closer the absolute distance between two phrases, the more related those two phrases are.

1.2.5.1.2 Definitions

Definition 1: The distance between two phrases ω_i and ω_j is defined to be zero ($\delta_{\omega_i \to \omega_j} := 0$) if the following conditions are met:

- ω_i appears in the set of phrases of the "definition" ($\eta^{A}_{(\omega_j)}$) of the article A_{ω_j} , and
- ω_j is in the set of hyperlinked phrases of the "definition" $(\Lambda^{A}_{(\omega_i)})$ of the article A_{ω_i}

$$\delta_{\omega_i \to \omega_j} := 0 \Leftrightarrow \omega_j \in \Lambda^{A}_{(\omega_i)} \ \ where \ \ \omega_i \in \eta^{A}_{(\omega_j)}$$

Definition 2: The distance from phrase ω_i to ω_j is defined to be one ($\delta_{\omega_i \to \omega_j} := 1$) if the following conditions are met:

- ω_i appears in the set of phrases of the "information box" ($\eta^{B}_{(\omega_j)}$) of the article A_{ω_j} , and
- ω_j is in the set of hyperlinked phrases of the "definition" $(\Lambda^{A}_{(\omega_i)})$ of the article A_{ω_i}

Definition 3: $\delta_{\omega_i \to \omega_j} := 2$ if the following conditions are met:

- ω_i appears in the set of phrases of the "information box" ($\eta^{B}_{(\omega_j)}$) of the article A_{ω_j} , and
- ω_j is in the set of hyperlinked phrases of the "information box" $(\Lambda^{B}_{(\omega_i)})$ of the article A_{ω_i}

Definition 4: $\delta_{\omega_i \to \omega_j} := 3$ if the following conditions are met:

- ω_i appears in the set of phrases of the "description" ($\eta^C_{(\omega_j)}$) of the article A_{ω_j} , and
- ω_j is in the set of hyperlinked phrases of the "definition" $(\Lambda^A_{(\omega_i)})$ of the article A_{ω_i}

Definition 5: $\delta_{\omega_i \to \omega_j} := 4$ if the following conditions are met:

- ω_i appears in the set of phrases of the "description" ($\eta^C_{(\omega_j)}$) of the article A_{ω_j} , and
- ω_j is in the set of hyperlinked phrases of the "information box" $(\Lambda^B_{(\omega_i)})$ of the article A_{ω_i}

Definition 6: $\delta_{\omega_i \to \omega_j} := 5$ if the following conditions are met:

- ω_i appears in the set of phrases of the "description" ($\eta^C_{(\omega_j)}$) of the article A_{ω_j} , and
- ω_j is in the set of hyperlinked phrases of the "description" $(\Lambda^C_{(\omega_i)})$ of the article A_{ω_i}

Definition 7: $\delta_{\omega_i \to \omega_j} := 6$ if the following conditions are met:

- ω_i appears in the set of phrases of the "summary of information" ($\eta^D_{(\omega_j)}$) of the article A_{ω_j} , and
- ω_j is in the set of hyperlinked phrases of the "definition" $(\Lambda^A_{(\omega_i)})$ of the article A_{ω_i}

Definition 8: $\delta_{\omega_i \to \omega_j} := 7$ if the following conditions are met:

- ω_i appears in the set of phrases of the "summary of information" ($\eta^D_{(\omega_j)}$) of the article A_{ω_j} , and
- ω_j is in the set of hyperlinked phrases of the "information box" $(\Lambda^B_{(\omega_i)})$ of the article A_{ω_i}

Definition 9: $\delta_{\omega_i \to \omega_j} := 8$ if the following conditions are met:

- ω_i appears in the set of phrases of the "summary of information" ($\eta^D_{(\omega_j)}$) of article A_{ω_j} , and
- ω_j is in the set of hyperlinked phrases of the "description" $(\Lambda^C_{(\omega_i)})$ of article A_{ω_i}

Definition 10: $\delta_{\omega_i \to \omega_j} := 9$ if the following conditions are met:

- ω_i does not exist in any of the sets $\eta^A_{(\omega_j)}$, $\eta^B_{(\omega_j)}$, or $\eta^C_{(\omega_j)}$ of the article A_{ω_j} ,
- ω_j is in the set of hyperlinked phrases $\Lambda^A_{(\omega_i)}$, $\Lambda^B_{(\omega_i)}$, or $\Lambda^C_{(\omega_i)}$ of the article A_{ω_i} , and
- ω_j is in the set of phrases of "categories" ($\eta^E_{(\omega_j)}$) of the article A_{ω_i}

Definition 11: $\delta_{\omega_i \to \omega_j} := 10$ if the following conditions are met:

- ω_i appears in the set of phrases of the “summary of information” ($\eta^{D}_{(\omega_j)}$) of article A_{ω_j} , and
- ω_j is in the set of hyperlinked phrases of the “summary of information” $(\Lambda^{D}_{(\omega_i)})$ of article A_{ω_i}

Definition 12: $\delta_{\omega_i \to \omega_j} := 11$ if the following conditions are met:

- ω_i does not exist in any of the sets $\eta^{A}_{(\omega_j)}$ of the article A_{ω_j} ,
- ω_j is in the set of hyperlinked phrases $\Lambda^{B}_{(\omega_i)}$ of the article A_{ω_i} , and

Definition 13: $\delta_{\omega_i \to \omega_j} := 12$ if the following conditions are met:

- ω_i does not exist in any of the sets $\eta^{B}_{(\omega_j)}$ of the article A_{ω_j} ,
- ω_j is in the set of hyperlinked phrases $\Lambda^{B}_{(\omega_i)}$ of the article A_{ω_i} , and

Definition 14: $\delta_{\omega_i \to \omega_j} := 13$ if the following conditions are met:

- ω_i does not exist in any of the sets $\eta^{C}_{(\omega_j)}$ of the article A_{ω_j} ,
- ω_j is in the set of hyperlinked phrases $\Lambda^{C}_{(\omega_i)}$ of the article A_{ω_i} , and

Definition 15: $\delta_{\omega_i \to \omega_j} := 14$ if the following conditions are met:

- ω_i does not exist in any of the sets $\eta^{D}_{(\omega_j)}$ of the article A_{ω_j} ,
- ω_j is in the set of hyperlinked phrases $\Lambda^{D}_{(\omega_i)}$ of the article A_{ω_i} , and

We define areas 0 to 14 ($\overset{0}{\Omega}_{(\omega)}, \overset{1}{\Omega}_{(\omega)}, \overset{2}{\Omega}_{(\omega)}, \ldots, \overset{14}{\Omega}_{(\omega)}$) to be the sets of phrases relevant to the phrase ω at distances 0 to 14.

$$\left(\overset{0}{\Omega}_{(\omega)} \cup \overset{1}{\Omega}_{(\omega)} \cup \overset{3}{\Omega}_{(\omega)} \cup \overset{6}{\Omega}_{(\omega)} \cup \overset{11}{\Omega}_{(\omega)} \right) \subseteq \Lambda^{A}_{(\omega)}$$

$$\left(\overset{2}{\Omega}_{(\omega)} \cup \overset{4}{\Omega}_{(\omega)} \cup \overset{7}{\Omega}_{(\omega)} \cup \overset{12}{\Omega}_{(\omega)}\right) \subseteq \Lambda^{B}_{(\omega)}$$

$$\left(\overset{5}{\Omega}_{(\omega)} \cup \overset{8}{\Omega}_{(\omega)} \cup \overset{13}{\Omega}_{(\omega)}\right) \subseteq \Lambda^{C}_{(\omega)}$$

$$\left(\overset{10}{\Omega}_{(\omega)} \cup \overset{14}{\Omega}_{(\omega)}\right) \subseteq \Lambda^{D}_{(\omega)}$$

$$\overset{9}{\Omega}_{(\omega)} \subseteq \left(\Lambda^{A}_{(\omega)} \cup \Lambda^{B}_{(\omega)} \cup \Lambda^{C}_{(\omega)}\right)$$

While phrases at distances 0 to 6 are assumed to be relevant to ω , there exist some phrases at distances 7 to 14 that are not. Thus, phrases in areas 7 to 14 shall go through a phase of further relevancy testing that can be summarized as follows.

For each phrase $\alpha_{(\omega)}$ in the areas (i.e. at distance) 7 to 14, if the number of phrases (in areas 0 to 13) matching in articles $\mathrm{A}_{\alpha(\omega)}$ is above a certain threshold T , then, $\alpha_{(\omega)}$ is considered relevant to ω .

$$\begin{aligned}
&\forall \alpha \in \overset{n}{\Omega}_{(\omega)}, 7 \le n \le 14, \\
&\alpha \text{ is relevant to } \omega \quad \text{if and only if } \left| \eta_{(\alpha)} \cap \left(\prod_{i=0}^{13} \overset{i}{\Omega}_{(\omega)} \right) \right| > T \\
&\textit{where } T = f(i,n,m) \textit{ and } m = \left| \prod_{n=7}^{14} \overset{n}{\Omega}_{(\omega)} \right|
\end{aligned}$$

Once all the definitions are applied to a Wikipedia article, a phrase fingerprint is generated (Figure 5).

1.2.5.1.3 Examples

Example 1: The word “iPhone” is in the set of hyperlinked phrases of the “definition” section $(\Lambda^{A}_{(Apple\ Inc.)})$ for the article $A_{Apple\ Inc.}$ describing “Apple Inc.” (Figure 7). Also, the phrase “Apple Inc.” appears in the set of phrases of the “definition” section ($\eta^{A}_{(iPhone)}$) for the article A_{iPhone} describing “iPhone” (Figure 8). We can conclude that the distance between the two phrases “Apple Inc.” and “iPhone” is zero as per definition 1.

$$\delta_{Apple\ Inc. \to iPhone} := 0$$

Example 2: The phrase “Personal computer” is in the set of hyperlinked phrases of the “definition” section $(\Lambda^{A}_{(Apple\ Inc.)})$ for the article $A_{Apple\ Inc.}$ describing “Apple Inc.” (Figure 7). Also, the alias of the phrase “Apple Inc.” appears in the set of phrases of the “description” section ($\eta^{C}_{(Personal\ computer)}$) for the article $A_{Personal\ computer}$ describing “Personal computer” (Figure 9). We can conclude that the distance from “Apple Inc.” to “Personal computer” is three as per definition 4.

$$\delta_{Apple\ Inc. \to Personal\ computer} := 3$$

Example 3: The phrase “Computer software” is in the set of hyperlinked phrases of the “definition” section $(\Lambda^{A}_{(Apple\ Inc.)})$ for the article $A_{Apple\ Inc.}$ describing “Apple Inc.” (Figure 7). Also, the alias of the phrase “Apple Inc.” appears in the set of phrases of the “description” section ($\eta^{C}_{(Computer\ software)}$) for the article $A_{Computer\ software}$ describing “Computer software” (Figure 10). We can conclude that the distance from “Apple Inc.” to “Computer software” is six as per definition 7.

$$\delta_{Apple\ Inc. \to Computer\ software} := 6$$

Example 4: The phrase “Apple Inc.” is in the set of hyperlinked phrases of the “definition” section $(\Lambda^{A}_{(Steve\ Jobs)})$ for the article $A_{Steve\ Jobs}$ describing “Steve Jobs” (Figure 11). Also, the phrase “Steve Jobs” appears in the set of phrases of the “information box” section ($\eta^{B}_{(Apple\ Inc.)}$) for the article $A_{Apple\ Inc.}$ describing “Apple Inc.” (Figure 7). We can conclude that the distance from “Steve Jobs” to “Apple Inc.” is one as per definition 2.

$$\delta_{Steve\ Jobs \to Apple\ Inc.} := 1 \qquad \Leftrightarrow \qquad \delta_{Apple\ Inc. \to Steve\ Jobs} := -1$$

1.3 nanobird Phrase Tokenizer

1.3.1 Definition

The *nanobird Phrase Tokenizer* is responsible for splitting an inputted text into sets of phrases. The tokenizer dissects an inputted text and generates three different sets of phrases S_U, S_A, S_L. The phrases in those sets consist of longest exact matches found in any of the unambiguous phrase, ambiguous phrase, or phrase aliases dictionaries (D_U, D_A, D_L). Those sets are, at a later stage, used by the *nanobird Context Analyzer* (as depicted in Figure 13) to analyze the relation among phrases in those sets and study their contextual correlation.

1.3.2 Algorithm

An inputted text is first split into an ordered set of words S_O using the space character as a divider between different words.

The *nanobird Phrase Tokenizer*'s algorithm starts by forming a phrase P from the first word ω_1, located at position i, in the ordered set of words S_O :

$$i \leftarrow 1 \ , \ P \leftarrow \omega_i \ , \ \omega_\mathrm{i} \in S_O$$

The phrase P is tested against the following four conditions and different actions are taken.

<u>Condition 1</u>: if an exact match of P is found in any of the dictionaries D_U, D_A, D_L, then the phrase is temporarily saved for future use ($P_{saved} \leftarrow P$) and the position of the last word used is temporarily saved in i_{saved}, i.e. $i_{saved} \leftarrow i$. Moreover, a new phrase ($P \leftarrow P \oplus \omega_{i+1}$) is formed by concatenating the next word ω_{i+1} with the current phrase P. The new phrase is then tested again against the four conditions.

<u>Condition 2</u>: if a submatch of P is found in any of the dictionaries D_U, D_A, D_L, then a new phrase ($P \leftarrow P \oplus \omega_{i+1}$) is formed by concatenating the next word ω_{i+1} with the current phrase P. The new phrase is then tested again against the four conditions. The position of the next word to continue processing from becomes $i \leftarrow i+1$

<u>Condition 3</u>: if neither an exact match nor a submatch of P is found in any of the dictionaries D_U, D_A, D_L and there exists a temporarily saved phrase P_{saved} (as described in condition 1) such that $P_{saved} \neq \varnothing$, then P_{saved} is considered to be the longest matching phrase found in any of the dictionaries D_U, D_A, D_L and thus, is added to one of sets of phrases S_U, S_A, or S_L depending on whether the exact match was found in D_U, D_A, or D_L respectively.

$$P_{saved} \in D_U \quad \Rightarrow \quad S_U \leftarrow S_U \cup \{P_{saved}\}$$

$$P_{saved} \in D_A \quad \Rightarrow \quad S_A \leftarrow S_A \cup \{P_{saved}\}$$

$$P_{saved} \in D_L \quad \Rightarrow \quad S_L \leftarrow S_L \cup \{P_{saved}\}$$

At this point, more actions are taken:

- P_{saved} is deleted ($P_{saved} \leftarrow \varnothing$)
- The position of the next word to continue processing from is learned ($i \leftarrow i_{saved} + 1$)
- The new phrase is now $P \leftarrow \omega_i$ and is tested against the four conditions.

Condition 4: if neither an exact match nor a submatch of P is found in any of the dictionaries (D_U, D_A, D_L) and there exists no temporarily saved phrase P_{saved} (as described in condition 1) that is $P_{saved} = \varnothing$ then:

- The position of the next word to continue processing from is learned ($i \leftarrow i_{saved} + 1$)
- The new phrase is now $P \leftarrow \omega_i$ and is tested against the four conditions.

Figure 12 is a pseudo-code of the algorithm that extracts the sets of longest matching phrases.

1.3.3 Example

Consider the following given text:

> "Steven Jobs requested that the Apple store on Chestnut Street in San Francisco be closed next Monday."

Also, assume the following predefined dictionaries:

D_U = {Apple Store, San Francisco, San Francis, Steve Jobs, Apple store }

D_A = {Apple, Chestnut, San, Monday}

D_L = {Steven Jobs, Steven Paul Jobs}

T he *nanobird Phrase Tokenizer* firsts generates the ordered set of words S_O from the above given text.

For instance, an inputted text shown in Figure 14 will generate the following ordered set of words.

$$S_O = \begin{Bmatrix} Steven, Jobs, requested, that, the, Apple, store, on, Chestnut, \\ Street, in, San, Francisco, be, closed, next, Monday. \end{Bmatrix}$$

The tokenizer looks up the dictionaries D_U, D_A, D_L and finds a submatch in D_L. As per ***condition 2***, a new phrase is formed by concatenating the next word with the current phrase. Thus,

$$P \leftarrow Steven\ Jobs$$

The tokenizer looks up P in the dictionaries D_U, D_A, D_L and finds an exact match in D_L. As per ***condition 1***, then the phrase is temporarily saved for future use and the position of the last word used is temporarily saved in i_{saved}. A new phrase is formed by concatenating the next word:

$$P_{saved} \leftarrow Steven\ Jobs$$

$$i_{saved} \leftarrow 2$$

$$P \leftarrow Steven\ Jobs\ requested$$

The tokenizer looks up P in the dictionaries D_U, D_A, D_L and finds neither an exact match nor a submatch. Thus, as per ***condition 3***, P_{saved} is considered to be the longest matching phrase found in dictionary D_L, and thus, is added to the set of phrases S_L.

$$P_{saved} \in D_L \quad \Rightarrow \quad S_L \leftarrow \{Steven\ Jobs\}$$

P_{saved} is deleted.

$$P_{saved} \leftarrow \varnothing$$

The position of the next word to continue processing from is learned ($i \leftarrow i_{saved} + 1$)

$$i \leftarrow 3$$

The new phrase is now:

$$P \leftarrow requested$$

A new lookup is done on P in the dictionaries D_U, D_A, D_L and neither an exact match nor a submatch is found and there exists no temporarily saved phrase that is $P_{saved} = \varnothing$. As per ***condition 4***, the position of the next word to continue processing from is learned ($i \leftarrow i_{saved} + 1$)

$$i \leftarrow 4$$

The new phrase is now:

$$P \leftarrow that$$

The processing of the text continues until the three sets S_U, S_A, S_L are extracted. For the above given text, the three generated sets are:

S_U = {Apple store, San Francisco }
S_A = {Chestnut, Monday}
S_L = {Steven Jobs }

1.4 nanobird Context Analyzer

1.4.1 Definition

The *nanobird Context Analyzer* is responsible for generating the *nanobird Context Graph*, a graph depicting the relations between the different phrases extracted by the *nanobird Phrase Tokenizer*. The graph's vertices consist of phrases, while its edges represent the distances between phrases. By looking at the graph and the correlation between the different phrases extracted from a given text, it becomes possible to learn the most important phrases that describe/outline/summarize that given text.

1.4.2 Algorithm

The *nanobird Context Analyzer* takes, as an input, the sets of phrases S_U, S_A, S_L created by passing an inputted natural language data (text) to the *nanobird Phrase Tokenizer*.

Each phrase in each set is then filtered out when compared against the sets of keys of the stop phrases dictionary (K_{D_U}). Any phrase matching any of the keys in K_{D_U} is removed from the three sets. This results in new sets of phrases S'_U, S'_A, and S'_L.

The resulting set of ambiguous phrases ($S_A^{'}$) is then converted into a set of unambiguous phrases (S_U^A) using dictionary D_A .

Similarly, the set of phrase aliases ($S_L^{'}$) is then converted into a set of unambiguous phrases (S_U^L) using dictionary D_L . Then, the three sets of unambiguous phrases are merged into a single set S .

$$S = S_U^{'} \cup S_U^A \cup S_U^L$$

Every pair of phrases from the set S is looked up in the fingerprint dictionary D_F and the distance between the pair of phrases is extracted. Once all the distances are extracted between every pair of phrases, a graph is generated by the *nanobird Context Analyzer*. The *nanobird Context Analyzer* internal flowchart is shown in Figure 13.

1.4.3 Example

The following example goes step by step through this context analysis process.

An example of a natural language data is given in Figure 14. The three sets of phrases generated by passing the text through the *nanobird Phrase Tokenizer* (as described in section 1.3) are:

$$S_U = \{\text{Climate change}\}$$

$$S_A = \{\text{atmosphere, Wednesday}\}$$

$$S_L = \{\text{climate crisis, greenhouse gases, ecosystems}\}$$

Assume the following to be a predefined stop phrases dictionary:

$$D_S = \{\text{who, best, where, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday, once}\}$$

The three sets are first filtered using the stop phrases dictionary D_S , yielding the following sets where the word "Wednesday" is omitted since it is part of stop phrases dictionary.

$$S_U^{'} = \{\text{Climate change}\}$$

$$S_A^{'} = \{\text{atmosphere}\}$$

$$S_L^{'} = \{\text{climate crisis, greenhouse gases, ecosystems}\}$$

The new set $S_A^{'}$ is then transformed using dictionaries D_A (refer to Figure 3) into the set S_A^U . For instance, the word "*atmosphere*" is replaced with the phrases *"Atmosphere of Earth", "Atmosphere",* *"Extraterrestrial atmospheres", "Stellar atmosphere", "Gas", "Adobe Atmosphere", and "Atmosphere Visual Effects".*

Similarly, the new set S_L' is then transformed using dictionaries D_L (refer to Figure 4) into the set S_L^U. For instance, the phrase "*climate crisis*" is replaced with the phrase *"Global warming",* while the phrase *"ecosystems"* is replaced with the phrase *"Ecosystem".*

The new resulting sets are thus:

$$S_U' = \{\text{Climate change}\}$$

$$S_A^U = \{\text{Atmosphere of Earth, Atmosphere,} \Box \text{, Adobe Atmosphere, Atmosphere Visual Effects}\}$$

$$S_L^U = \{\text{Global warming, Greenhouse gas, Ecosystem}\}$$

The three sets of unambiguous phrases are merged into a single set S.

$$S = \left\{\begin{array}{l}\text{Climate change,}\\ \text{Atmosphere of Earth,}\\ \text{Atmosphere,}\\ \Box\text{,}\\ \text{Adobe Atmosphere,}\\ \text{Atmosphere Visual Effects,}\\ \text{Global warming,}\\ \text{Greenhouse gas,}\\ \text{Ecosystem}\end{array}\right\}$$

Using the dictionary of phrase fingerprints D_F, the phrases in the set S are correlated. Figure 15 depicts a fingerprint map of the phrase "Atmosphere of Earth". As shown in Figure 15, the phrase "Atmosphere of Earth" is related to the phrase "Climate change" positioned at a distance 10.

$$\delta_{\text{Atmosphere of Earth} \to \text{Climate change}} := 10$$

In Figure 16, a relation exists between the two words "Climate change" and "Global warming", such that:

$$\delta_{\text{Climate change} \to \text{Global warming}} := -3 \Leftrightarrow \delta_{\text{Global warming} \to \text{Climate change}} := 3$$

Similarly, the maps of phrase fingerprints depicted in Figure 17, Figure 18, and Figure 19 yield the following relations:

$$\delta_{\text{Ecosystem} \to \text{Global warming}} := 10$$

$$\delta_{\text{Global warming} \to \text{Atmosphere}} := 14$$

$$\delta_{\text{Global warming} \rightarrow \text{Greenhouse gas}} := 6$$

$$\delta_{\text{Greenhouse gas} \rightarrow \text{Atmosphere of Earth}} := 6$$

On the other hand, no relations were found between the phrase "Adobe Atmosphere" and the other phrases of the set S. As a result, the phrase "*Adobe Atmosphere*" is discarded. Finally, a *nanobird Context Graph* is generated (Figure 20) showing only the phrases for which correlations with other phrases are found using the fingerprints. Those phrases describe/outline/summarize the natural language data of Figure 14.

2 nRE Applications

The application of nRE is very rich, ranging from profiling social media users to delivering tailored advertisements. The following sections summarize some of the applications where nRE could play a big part.

2.1 Twitter and Other Social Networking Sites

Pinpointing interests of social media (Facebook, Twitter, and Google+) users has been considered to be a very challenging problem. Using nRE, profiling users has been proven to be feasible. The solution opens the door to predicting what a user will want to read within their social media feeds and what news articles he/she will be interested in reading in the future.

A Twitter application using nRE has been developed where twitter accounts are processed and profiled. For each account, the last 100 tweets are collected and each tweet is passed to nRE to yield the set of phrases that best describe the tweets or the links within the tweets. Then, the occurrence of the same phrases in all the 100 sets (generated by all the tweets) is noted down and the phrases with the highest number of occurrences are deemed to represent the user's main interests. If a twitter user does not share links/tweets, the twitter accounts they are following are profiled and the user is considered to have the collective interests of the users he/she is following.

The nanobird twitter application is able to directly spot the interests of every single twitter user and recommend other twitter users sharing common interests (Figure 21). The application is also able to filter out the non-relevant links that are coming up in a user's feed. Massive amount of noise in users account could be turned into a fine tune or pitch of tweets/links that are relevant to the users' interests.

Similarly, the nanobird social media application profiles users on other social networking sites (e.g. Facebook, Google+) and determines their interest by processing the links and text they share.

2.2 News

The nanobird news application's main focus is on providing users with news tailored to their likings. The news application has three main components: a profiling component, a news context analyzer, a news delivery component.

- Profiling component: Learning a user's news interests could be achieved in two ways; either by conducting an initial profiling of the provided user's social media accounts, or by learning, over time, from the news the user reads on the nanobird news application. The latter learning technique is based on monitoring a user's reading behavior and some aging algorithms to clean out entities that are no longer relevant to a user and entities that were only part of a sudden world event or non-timeless event.

- News context analyzer: As fresh news articles are published, the nRE profiles every new article and extracts the set of words that best describes the article.

- News delivery component: By combining the results from the profiling component and the news context analyzer component, it becomes possible to deliver news closest to the users' interests.

2.3 Movie Recommendations

The nanobird movie application is an application that provides movie recommendation to users. It comprises two main components: an movie understanding component and recommendation component.

- Movie understanding component: Every film's plot synopsis and summary is passed through the nRE where the relevant entities of the movie are extracted and classified.

- Recommendations component: A user is asked to input a movie of their preference and the recommendation component provides him/her with the closest matching movie in terms of synopsis and/or genre such as "spy action thriller". The nanobird movie application will always recommend more recent films (including the latest films out on DVD, in theaters, or coming soon) before the outdated ones.

2.4 Book Suggestions

The nanobird book recommendation application is similar to the nanobird movie recommendations one except that the profiling of a book is achieved by passing the story plot/summary to the nRE.

2.5 Advertisement

The nanobird advertisement application helps content providers increase their revenue from advertisements by displaying relevant ads on their published sites. Relevant ads are displayed based on the site-visitor's interests (extracted from his/her social media accounts as previously described) or based on the content of the pages itself using nRE.

2.6 Search

By understanding text just as a human does, the nanobird search application is able to organize all the content of the web just as a human would organize documents in a ling cabinet based on subject. A nanobird search application would be able to give search results based on relevancy to what a user searches for. The nanobird search application could be a plug-in to search engines such as Google and Bing to improve ranking of results and results classification.

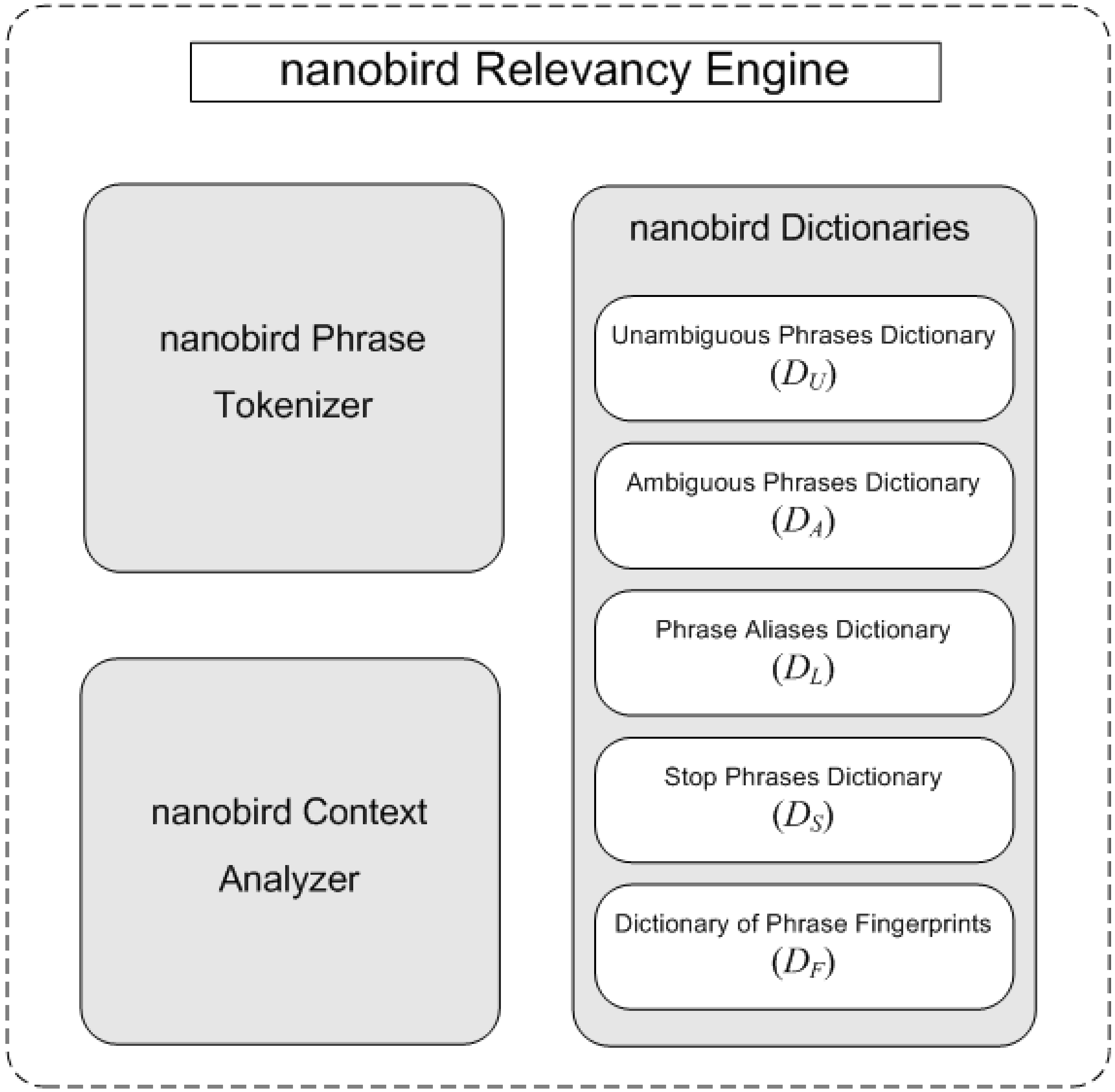


Figure 1. The nanobird Relevancy Engine Components Diagram

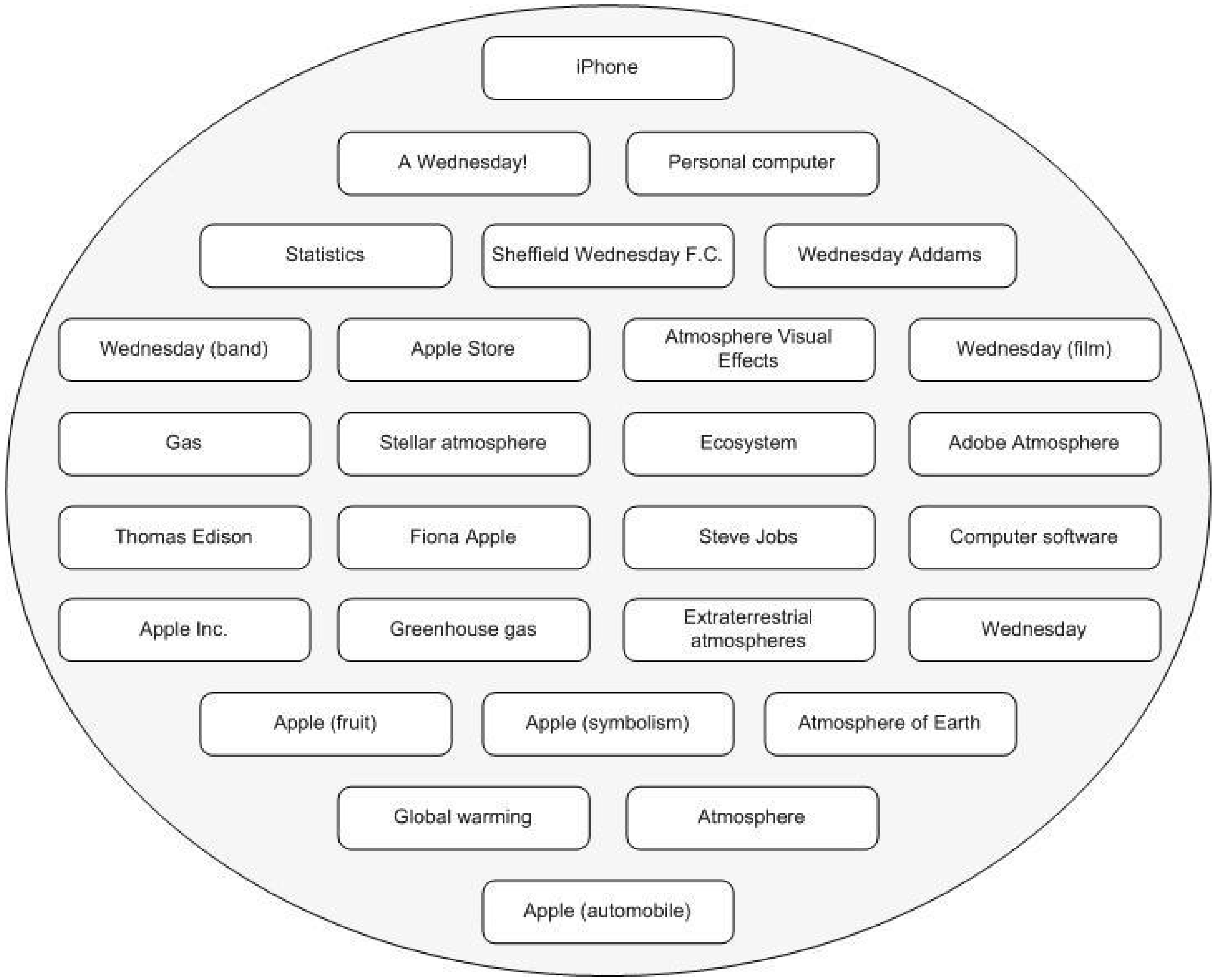

iPhone
A Wednesday!
Personal computer
Statistics
Sheffield Wednesday F.C.
Wednesday Addams
Wednesday (band)
Apple Store
Atmosphere Visual Effects
Wednesday (film)
Gas
Stellar atmosphere
Ecosystem
Adobe Atmosphere
Thomas Edison
Fiona Apple
Steve Jobs
Computer software
Apple Inc.
Greenhouse gas
Extraterrestrial atmospheres
Wednesday
Apple (fruit)
Apple (symbolism)
Atmosphere of Earth
Global warming
Atmosphere
Apple (automobile)

Figure 2. A subset of the unambiguous phrases dictionary

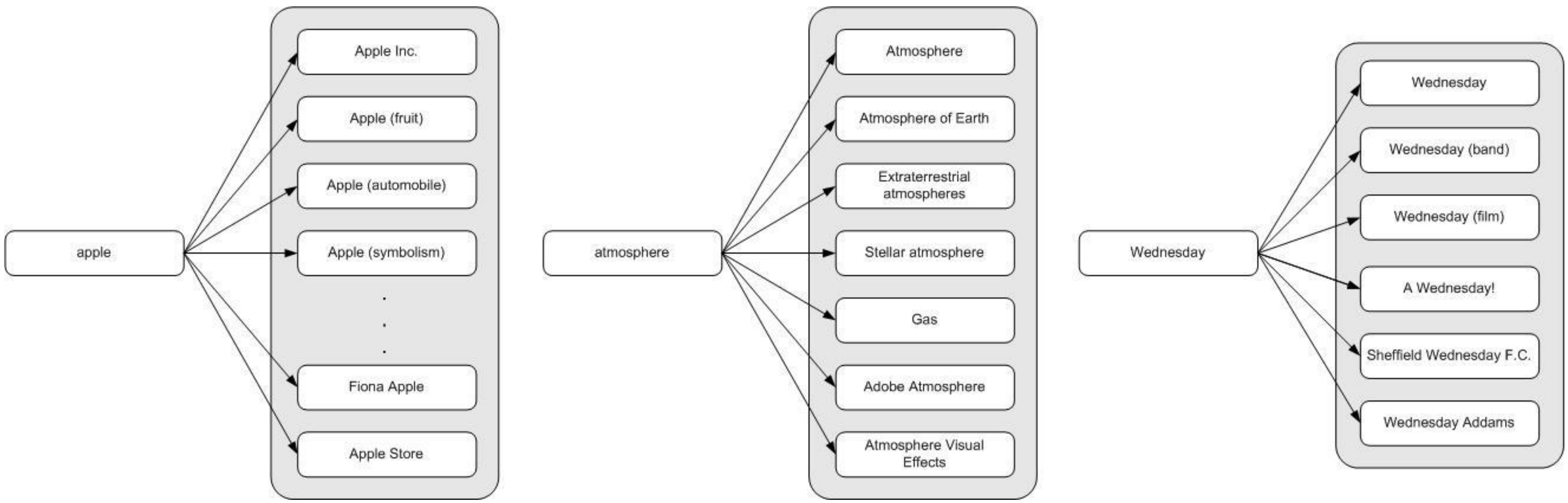


Figure 3. An example of ambiguous phrases (apple, atmosphere, Wednesday) and their mapping to unambiguous phrases

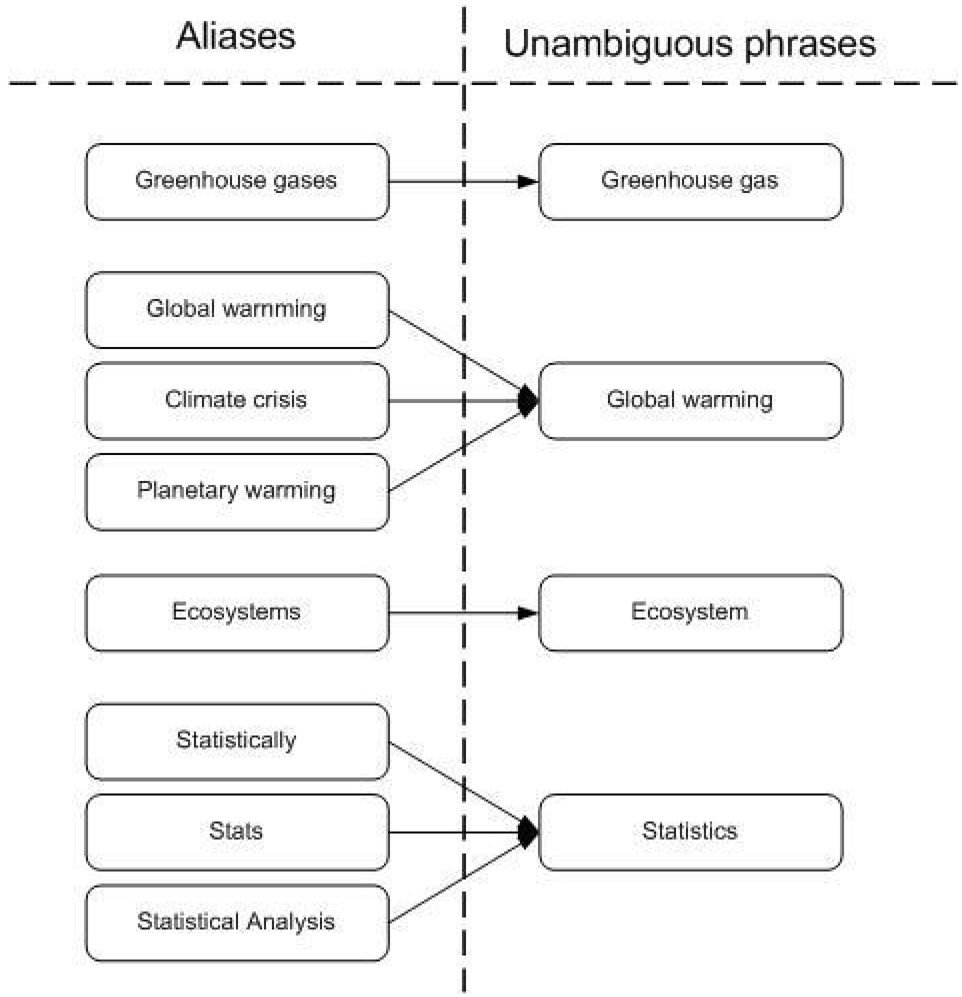


Figure 4. An example of phrase aliases and their mapping to unambiguous phrases

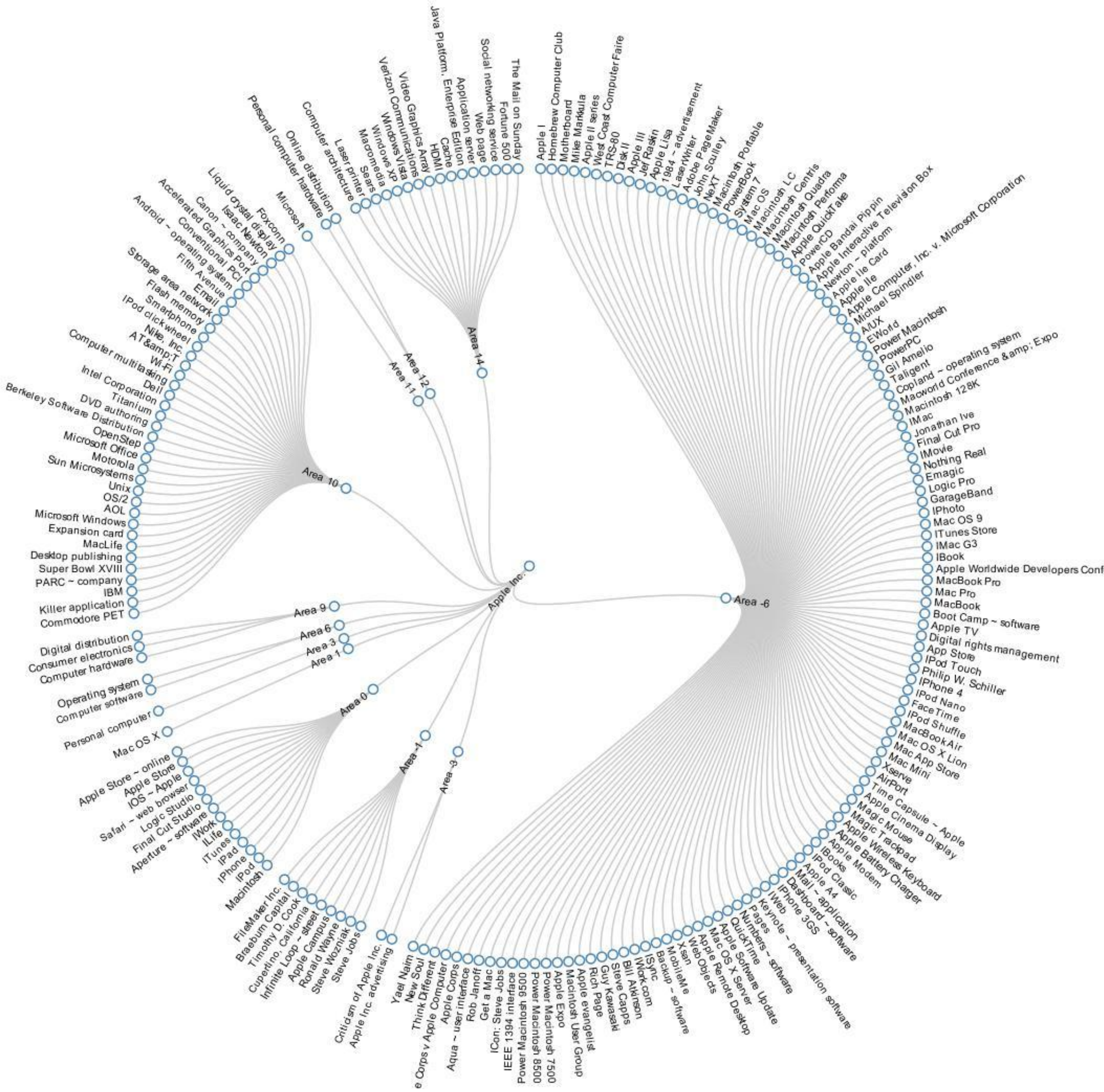


Figure 5. A fingerprint of the phrase “Apple Inc.”

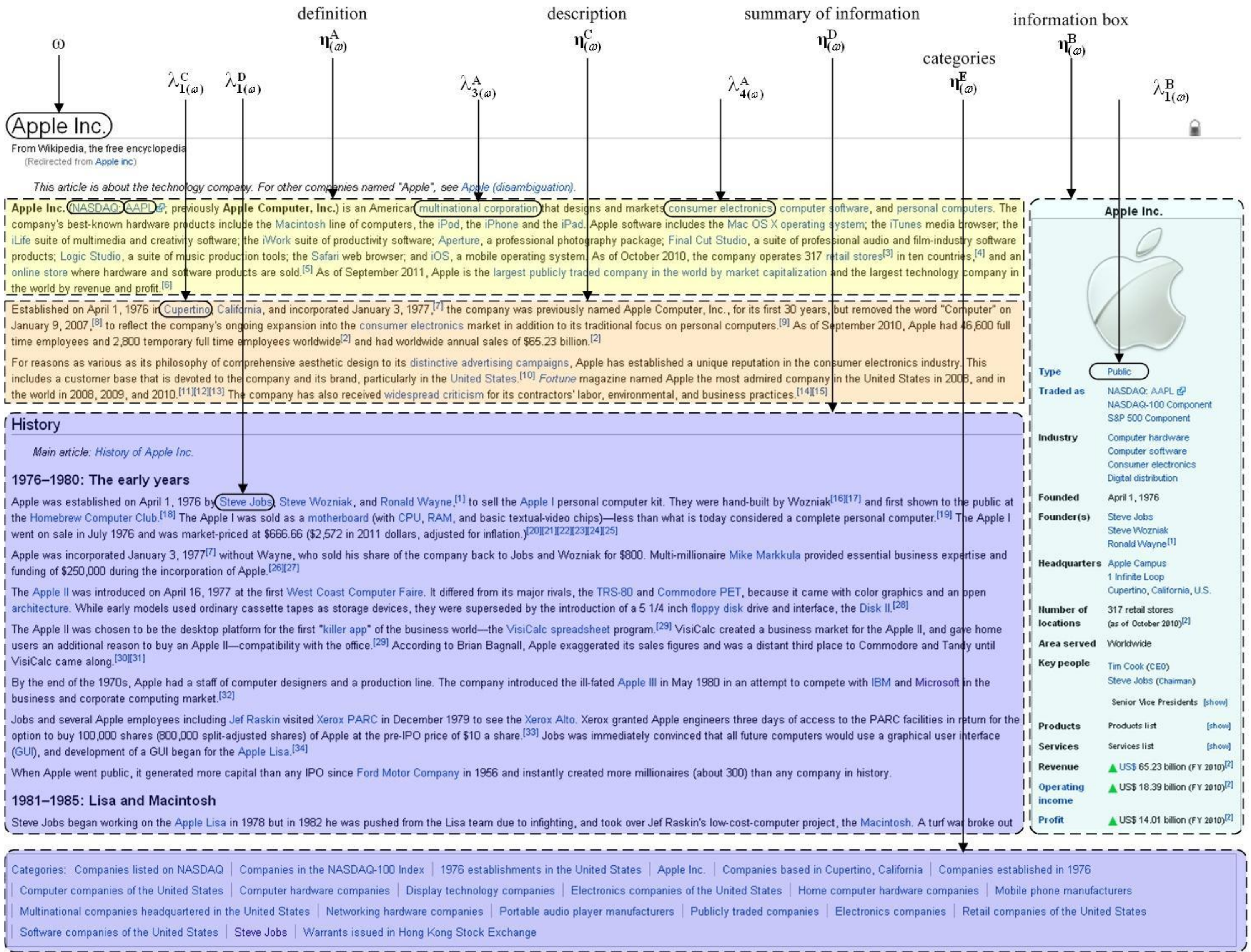

definition
description
summary of information
information box
categories
ω
$\eta^{A}_{(\omega)}$
$\eta^{C}_{(\omega)}$
$\eta^{D}_{(\omega)}$
$\eta^{B}_{(\omega)}$
$\eta^{E}_{(\omega)}$
$\lambda^{C}_{1(\omega)}$
$\lambda^{D}_{1(\omega)}$
$\lambda^{A}_{3(\omega)}$
$\lambda^{A}_{4(\omega)}$
$\lambda^{B}_{1(\omega)}$
Apple Inc.
From Wikipedia, the free encyclopedia
(Redirected from Apple inc)
This article is about the technology company. For other companies named "Apple", see Apple (disambiguation).
History
Main article: History of Apple Inc.
1976–1980: The early years
1981–1985: Lisa and Macintosh
Type Public
Traded as NASDAQ: AAPL NASDAQ-100 Component S&P 500 Component
Industry Computer hardware Computer software Consumer electronics Digital distribution
Founded April 1, 1976
Founder(s) Steve Jobs Steve Wozniak Ronald Wayne
Headquarters Apple Campus 1 Infinite Loop Cupertino, California, U.S.
Number of locations 317 retail stores (as of October 2010)
Area served Worldwide
Key people Tim Cook (CEO) Steve Jobs (Chairman)
Products Products list
Services Services list
Revenue US$ 65.23 billion (FY 2010)
Operating income US$ 18.39 billion (FY 2010)
Profit US$ 14.01 billion (FY 2010)
Categories: Companies listed on NASDAQ | Companies in the NASDAQ-100 Index | 1976 establishments in the United States | Apple Inc. | Companies based in Cupertino, California | Companies established in 1976 | Computer companies of the United States | Computer hardware companies | Display technology companies | Electronics companies of the United States | Home computer hardware companies | Mobile phone manufacturers | Multinational companies headquartered in the United States | Networking hardware companies | Portable audio player manufacturers | Publicly traded companies | Electronics companies | Retail companies of the United States | Software companies of the United States | Steve Jobs | Warrants issued in Hong Kong Stock Exchange

Figure 6. A Wikipedia article split into five defined sections

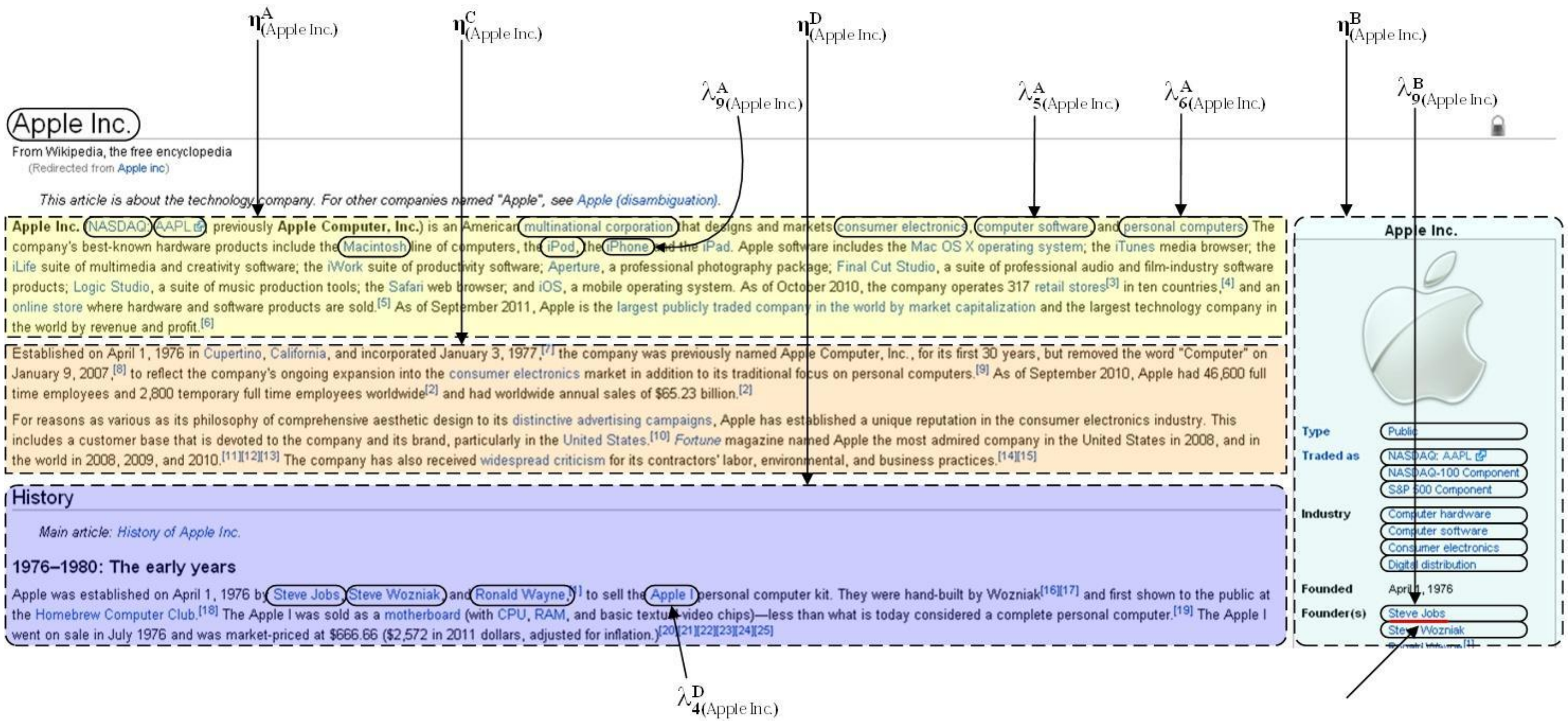
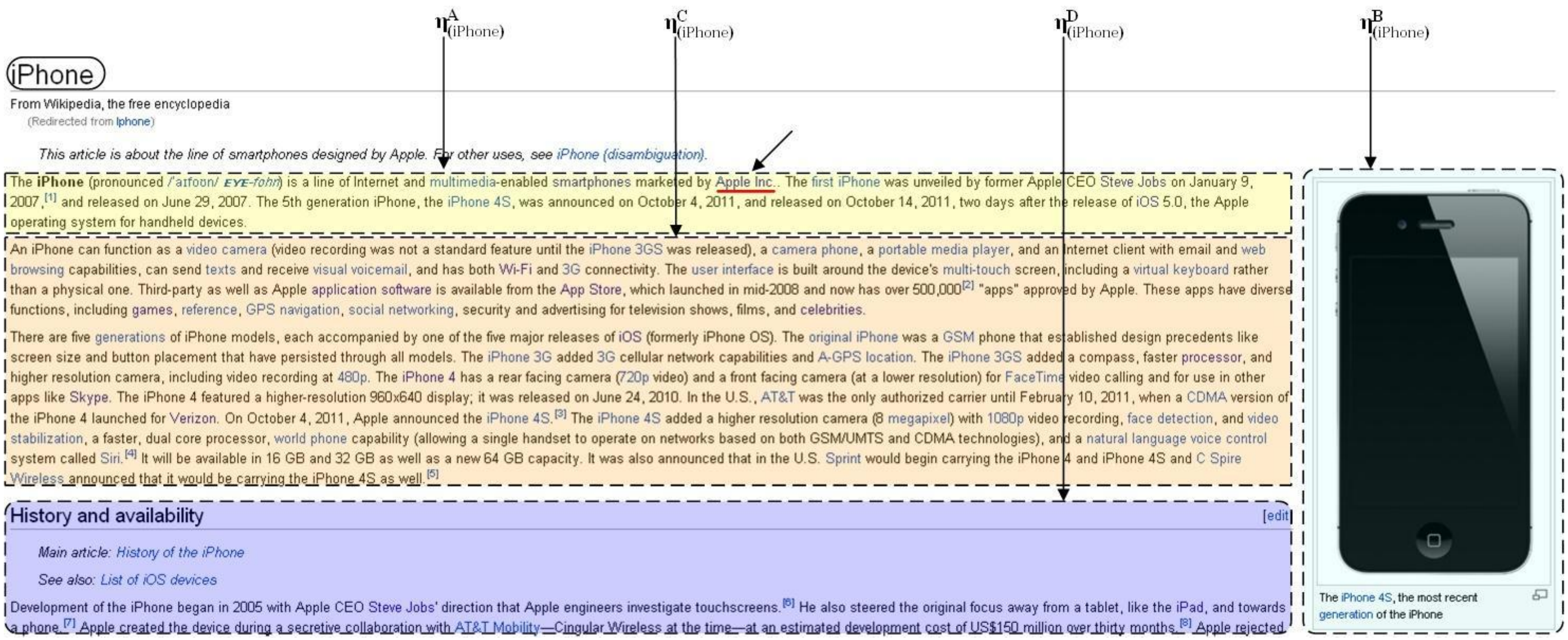


Figure 7. A Wikipedia article about "Apple Inc."



iPhone

From Wikipedia, the free encyclopedia

(Redirected from Iphone)

This article is about the line of smartphones designed by Apple. For other uses, see iPhone (disambiguation).

The **iPhone** (pronounced /ˈaɪfoʊn/ EYE-fohn) is a line of Internet and multimedia-enabled smartphones marketed by Apple Inc.. The first iPhone was unveiled by former Apple CEO Steve Jobs on January 9, 2007,[1] and released on June 29, 2007. The 5th generation iPhone, the iPhone 4S, was announced on October 4, 2011, and released on October 14, 2011, two days after the release of iOS 5.0, the Apple operating system for handheld devices.

An iPhone can function as a video camera (video recording was not a standard feature until the iPhone 3GS was released), a camera phone, a portable media player, and an Internet client with email and web browsing capabilities, can send texts and receive visual voicemail, and has both Wi-Fi and 3G connectivity. The user interface is built around the device's multi-touch screen, including a virtual keyboard rather than a physical one. Third-party as well as Apple application software is available from the App Store, which launched in mid-2008 and now has over 500,000[2] "apps" approved by Apple. These apps have diverse functions, including games, reference, GPS navigation, social networking, security and advertising for television shows, films, and celebrities.

There are five generations of iPhone models, each accompanied by one of the five major releases of iOS (formerly iPhone OS). The original iPhone was a GSM phone that established design precedents like screen size and button placement that have persisted through all models. The iPhone 3G added 3G cellular network capabilities and A-GPS location. The iPhone 3GS added a compass, faster processor, and higher resolution camera, including video recording at 480p. The iPhone 4 has a rear facing camera (720p video) and a front facing camera (at a lower resolution) for FaceTime video calling and for use in other apps like Skype. The iPhone 4 featured a higher-resolution 960x640 display; it was released on June 24, 2010. In the U.S., AT&T was the only authorized carrier until February 10, 2011, when a CDMA version of the iPhone 4 launched for Verizon. On October 4, 2011, Apple announced the iPhone 4S.[3] The iPhone 4S added a higher resolution camera (8 megapixel) with 1080p video recording, face detection, and video stabilization, a faster, dual core processor, world phone capability (allowing a single handset to operate on networks based on both GSM/UMTS and CDMA technologies), and a natural language voice control system called Siri.[4] It will be available in 16 GB and 32 GB as well as a new 64 GB capacity. It was also announced that in the U.S. Sprint would begin carrying the iPhone 4 and iPhone 4S and C Spire Wireless announced that it would be carrying the iPhone 4S as well.[5]

History and availability [edit]

Main article: History of the iPhone

See also: List of iOS devices

Development of the iPhone began in 2005 with Apple CEO Steve Jobs' direction that Apple engineers investigate touchscreens.[6] He also steered the original focus away from a tablet, like the iPad, and towards a phone.[7] Apple created the device during a secretive collaboration with AT&T Mobility—Cingular Wireless at the time—at an estimated development cost of US$150 million over thirty months.[8] Apple rejected

The iPhone 4S, the most recent generation of the iPhone

Figure 8. A Wikipedia article about "iPhone"

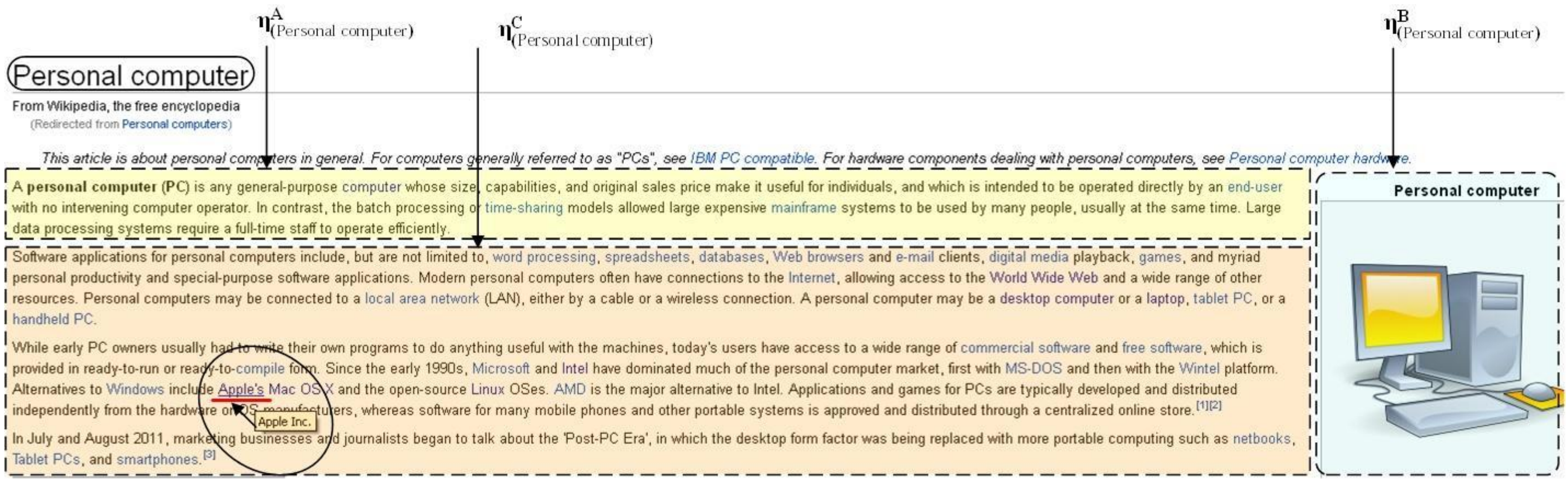
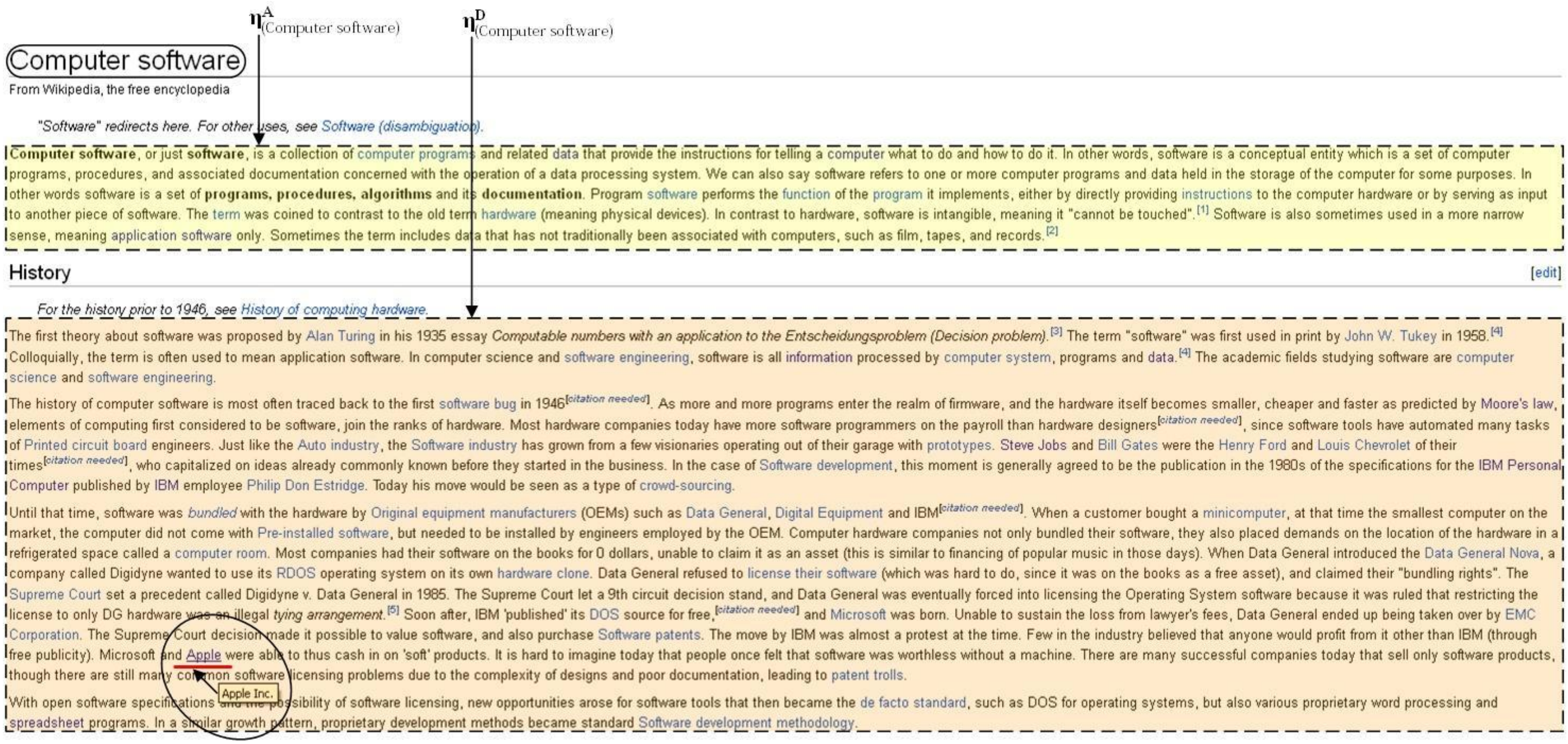


Figure 9. A Wikipedia article about "Personal computer"



$\eta^{A}_{(\text{Computer software})}$ $\eta^{D}_{(\text{Computer software})}$

Computer software

From Wikipedia, the free encyclopedia

"Software" redirects here. For other uses, see Software (disambiguation).

Computer software, or just **software**, is a collection of computer programs and related data that provide the instructions for telling a computer what to do and how to do it. In other words, software is a conceptual entity which is a set of computer programs, procedures, and associated documentation concerned with the operation of a data processing system. We can also say software refers to one or more computer programs and data held in the storage of the computer for some purposes. In other words software is a set of **programs, procedures, algorithms** and its **documentation**. Program software performs the function of the program it implements, either by directly providing instructions to the computer hardware or by serving as input to another piece of software. The term was coined to contrast to the old term hardware (meaning physical devices). In contrast to hardware, software is intangible, meaning it "cannot be touched".[1] Software is also sometimes used in a more narrow sense, meaning application software only. Sometimes the term includes data that has not traditionally been associated with computers, such as film, tapes, and records.[2]

History [edit]

For the history prior to 1946, see History of computing hardware.

The first theory about software was proposed by Alan Turing in his 1935 essay *Computable numbers with an application to the Entscheidungsproblem (Decision problem)*.[3] The term "software" was first used in print by John W. Tukey in 1958.[4] Colloquially, the term is often used to mean application software. In computer science and software engineering, software is all information processed by computer system, programs and data.[4] The academic fields studying software are computer science and software engineering.

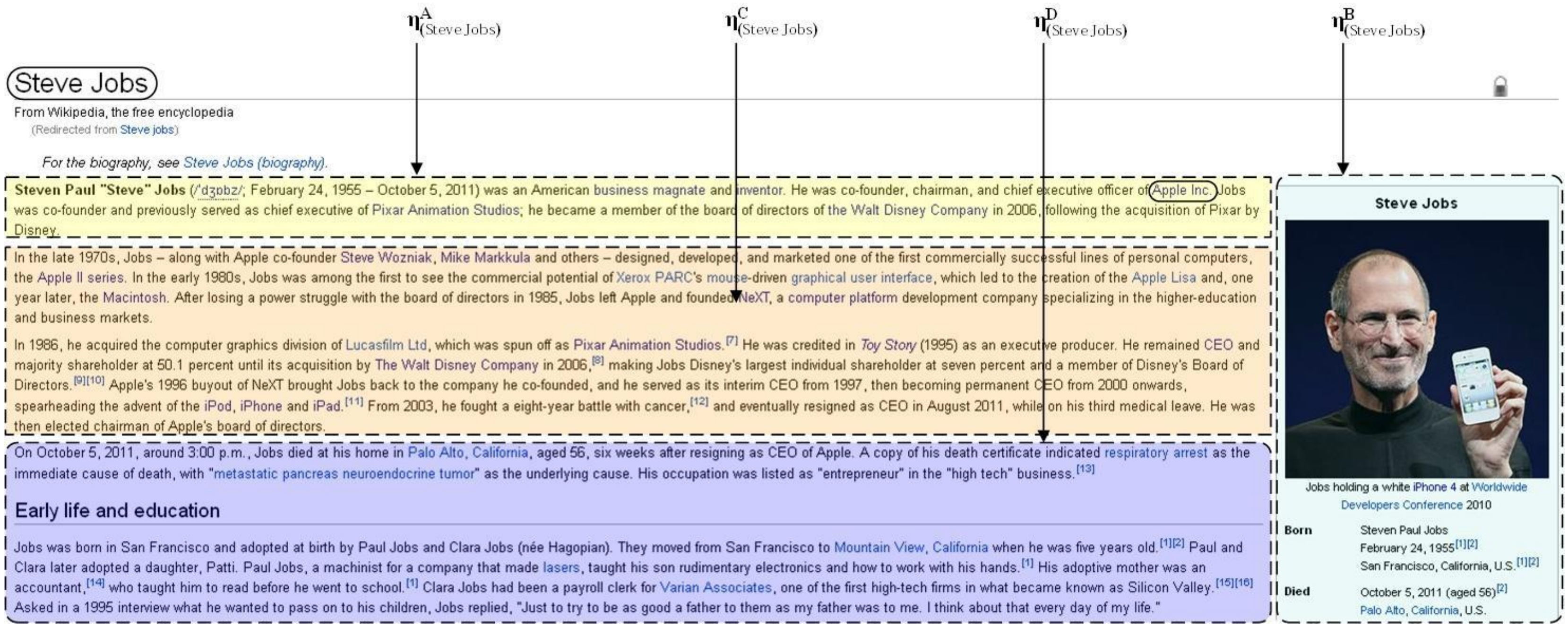
The history of computer software is most often traced back to the first software bug in 1946[citation needed]. As more and more programs enter the realm of firmware, and the hardware itself becomes smaller, cheaper and faster as predicted by Moore's law, elements of computing first considered to be software, join the ranks of hardware. Most hardware companies today have more software programmers on the payroll than hardware designers[citation needed], since software tools have automated many tasks of Printed circuit board engineers. Just like the Auto industry, the Software industry has grown from a few visionaries operating out of their garage with prototypes. Steve Jobs and Bill Gates were the Henry Ford and Louis Chevrolet of their times[citation needed], who capitalized on ideas already commonly known before they started in the business. In the case of Software development, this moment is generally agreed to be the publication in the 1980s of the specifications for the IBM Personal Computer published by IBM employee Philip Don Estridge. Today his move would be seen as a type of crowd-sourcing.

Until that time, software was *bundled* with the hardware by Original equipment manufacturers (OEMs) such as Data General, Digital Equipment and IBM[citation needed]. When a customer bought a minicomputer, at that time the smallest computer on the market, the computer did not come with Pre-installed software, but needed to be installed by engineers employed by the OEM. Computer hardware companies not only bundled their software, they also placed demands on the location of the hardware in a refrigerated space called a computer room. Most companies had their software on the books for 0 dollars, unable to claim it as an asset (this is similar to financing of popular music in those days). When Data General introduced the Data General Nova, a company called Digidyne wanted to use its RDOS operating system on its own hardware clone. Data General refused to license their software (which was hard to do, since it was on the books as a free asset), and claimed their "bundling rights". The Supreme Court set a precedent called Digidyne v. Data General in 1985. The Supreme Court let a 9th circuit decision stand, and Data General was eventually forced into licensing the Operating System software because it was ruled that restricting the license to only DG hardware was an illegal *tying arrangement*.[5] Soon after, IBM 'published' its DOS source for free,[citation needed] and Microsoft was born. Unable to sustain the loss from lawyer's fees, Data General ended up being taken over by EMC Corporation. The Supreme Court decision made it possible to value software, and also purchase Software patents. The move by IBM was almost a protest at the time. Few in the industry believed that anyone would profit from it other than IBM (through free publicity). Microsoft and Apple were able to thus cash in on 'soft' products. It is hard to imagine today that people once felt that software was worthless without a machine. There are many successful companies today that sell only software products, though there are still many common software licensing problems due to the complexity of designs and poor documentation, leading to patent trolls.

Apple Inc.

With open software specifications and the possibility of software licensing, new opportunities arose for software tools that then became the de facto standard, such as DOS for operating systems, but also various proprietary word processing and spreadsheet programs. In a similar growth pattern, proprietary development methods became standard Software development methodology.

Figure 10. A Wikipedia article about "Computer software"



Steve Jobs

From Wikipedia, the free encyclopedia

(Redirected from Steve jobs)

For the biography, see Steve Jobs (biography).

Steven Paul "Steve" Jobs (/ˈdʒɒbz/; February 24, 1955 – October 5, 2011) was an American business magnate and inventor. He was co-founder, chairman, and chief executive officer of Apple Inc. Jobs was co-founder and previously served as chief executive of Pixar Animation Studios; he became a member of the board of directors of the Walt Disney Company in 2006, following the acquisition of Pixar by Disney.

In the late 1970s, Jobs – along with Apple co-founder Steve Wozniak, Mike Markkula and others – designed, developed, and marketed one of the first commercially successful lines of personal computers, the Apple II series. In the early 1980s, Jobs was among the first to see the commercial potential of Xerox PARC's mouse-driven graphical user interface, which led to the creation of the Apple Lisa and, one year later, the Macintosh. After losing a power struggle with the board of directors in 1985, Jobs left Apple and founded NeXT, a computer platform development company specializing in the higher-education and business markets.

In 1986, he acquired the computer graphics division of Lucasfilm Ltd, which was spun off as Pixar Animation Studios.[7] He was credited in *Toy Story* (1995) as an executive producer. He remained CEO and majority shareholder at 50.1 percent until its acquisition by The Walt Disney Company in 2006,[8] making Jobs Disney's largest individual shareholder at seven percent and a member of Disney's Board of Directors.[9][10] Apple's 1996 buyout of NeXT brought Jobs back to the company he co-founded, and he served as its interim CEO from 1997, then becoming permanent CEO from 2000 onwards, spearheading the advent of the iPod, iPhone and iPad.[11] From 2003, he fought a eight-year battle with cancer,[12] and eventually resigned as CEO in August 2011, while on his third medical leave. He was then elected chairman of Apple's board of directors.

On October 5, 2011, around 3:00 p.m., Jobs died at his home in Palo Alto, California, aged 56, six weeks after resigning as CEO of Apple. A copy of his death certificate indicated respiratory arrest as the immediate cause of death, with "metastatic pancreas neuroendocrine tumor" as the underlying cause. His occupation was listed as "entrepreneur" in the "high tech" business.[13]

Early life and education

Jobs was born in San Francisco and adopted at birth by Paul Jobs and Clara Jobs (née Hagopian). They moved from San Francisco to Mountain View, California when he was five years old.[1][2] Paul and Clara later adopted a daughter, Patti. Paul Jobs, a machinist for a company that made lasers, taught his son rudimentary electronics and how to work with his hands.[1] His adoptive mother was an accountant,[14] who taught him to read before he went to school.[1] Clara Jobs had been a payroll clerk for Varian Associates, one of the first high-tech firms in what became known as Silicon Valley.[15][16] Asked in a 1995 interview what he wanted to pass on to his children, Jobs replied, "Just to try to be as good a father to them as my father was to me. I think about that every day of my life."

Steve Jobs

Jobs holding a white iPhone 4 at Worldwide Developers Conference 2010

Born	Steven Paul Jobs February 24, 1955[1][2] San Francisco, California, U.S.[1][2]
Died	October 5, 2011 (aged 56)[2] Palo Alto, California, U.S.

Figure 11. A Wikipedia article about "Steve Jobs"

```
Given:
    D denotes the set of dictionaries (D_U, D_A, D_L)
    p denotes a phrase
    G is an inputted text consisting of multiple words
    S is a set of longest matching phrases such that S ⊂ D
    ⊕ is the operation of joining two phrases using whitespace

Input:
    G = w_1 ⊕ w_2 ⊕ … ⊕ w_n

Algorithm:
    g ← split G into set of words
    p ← Ø
    S ← Ø
    for i = 1 to n
        p ← p ⊕ w_i
        if s is a match in D then
            p_saved ← p
            i_saved ← i
            continue
        else if p is a submatch  in D then
            continue
        else if p_saved ≠ Ø then
            S ← S ∪ {p_saved}
        p ← Ø
        p_saved ← Ø
        i ← i_saved + 1
```

Figure 12. nanobird Phrase Tokenizer Algorithm

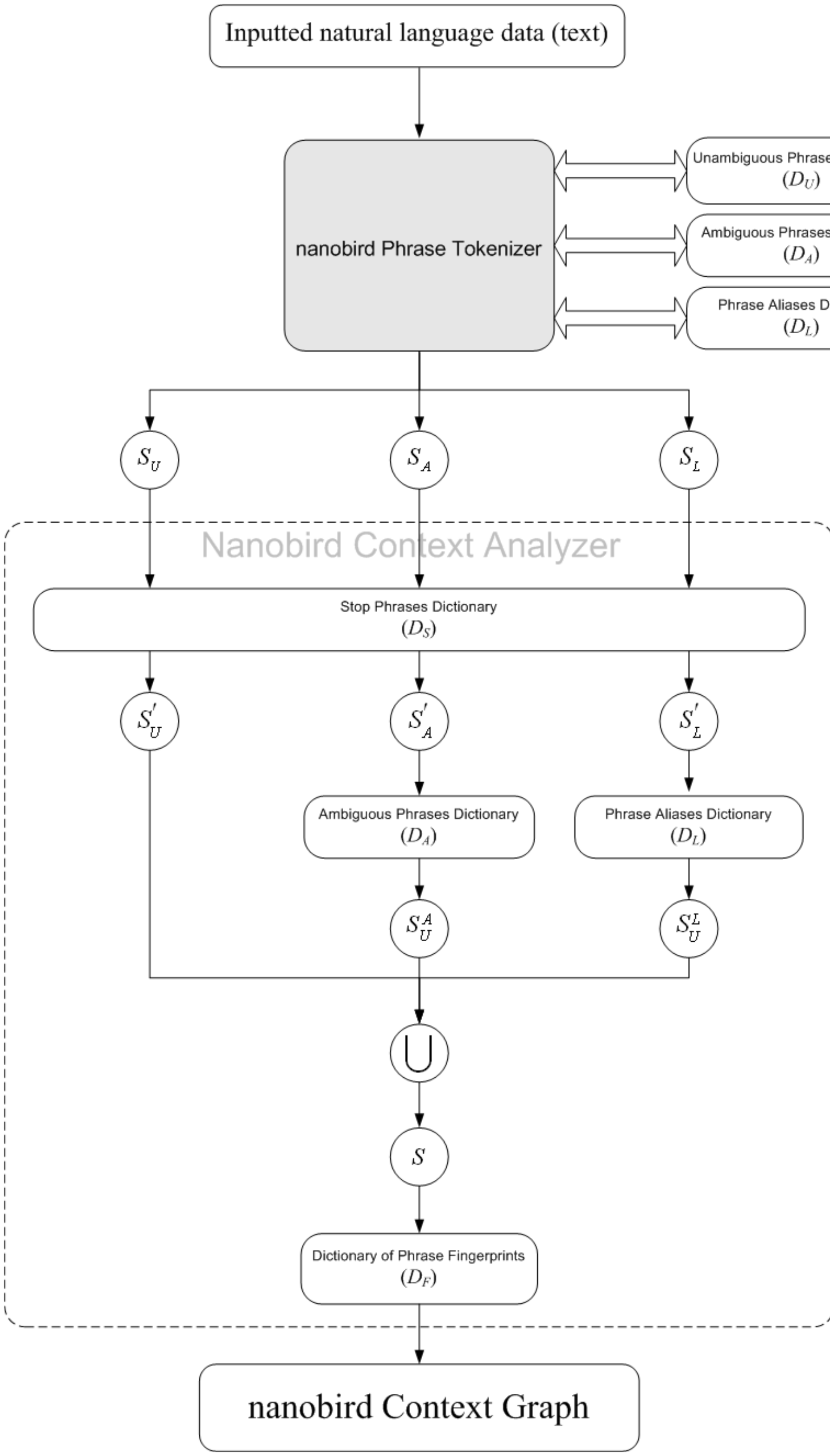
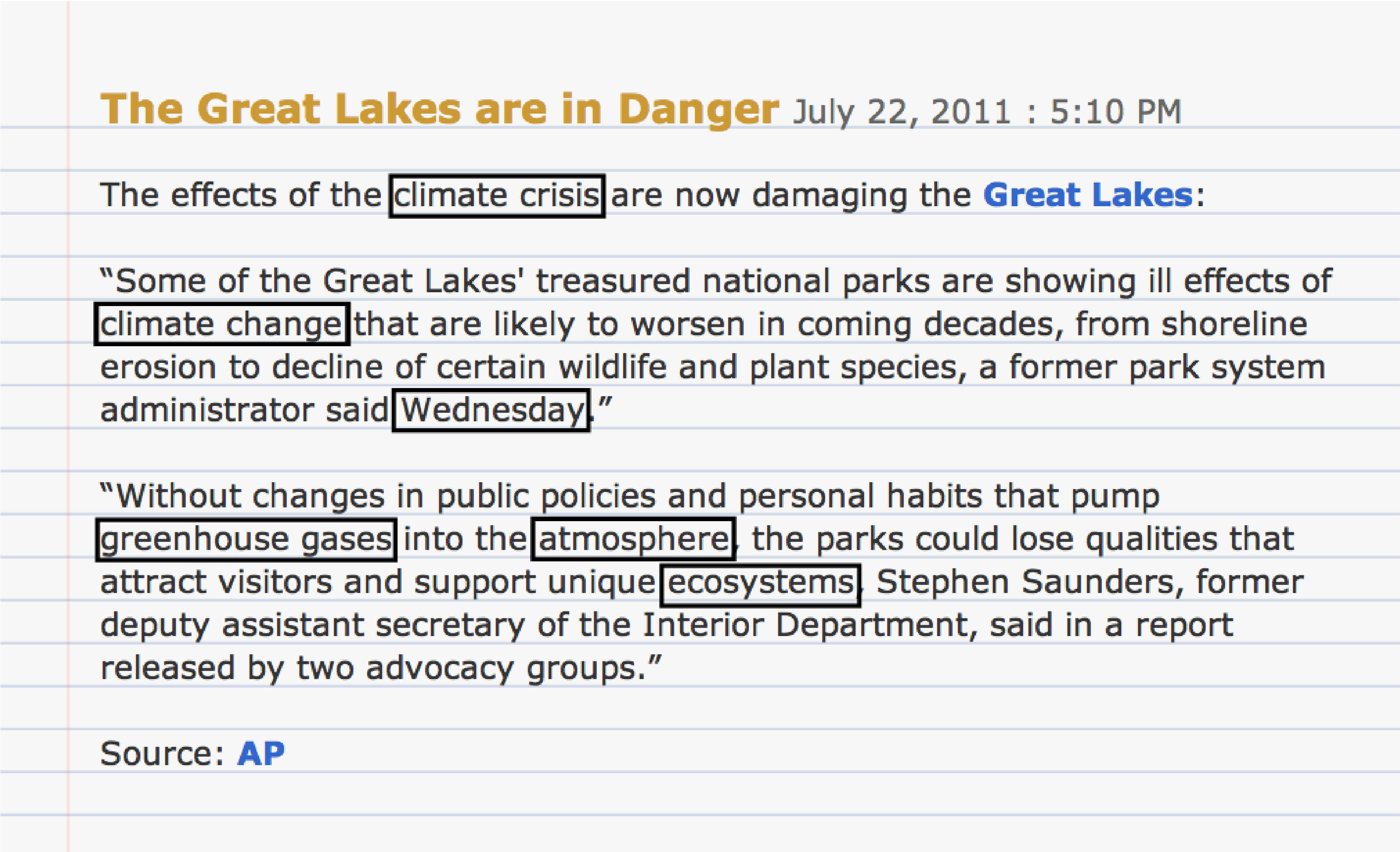


Figure 13. nanobird Relevancy Engine Flowchart



The Great Lakes are in Danger July 22, 2011 : 5:10 PM

The effects of the climate crisis are now damaging the **Great Lakes**:

"Some of the Great Lakes' treasured national parks are showing ill effects of climate change that are likely to worsen in coming decades, from shoreline erosion to decline of certain wildlife and plant species, a former park system administrator said Wednesday."

"Without changes in public policies and personal habits that pump greenhouse gases into the atmosphere, the parks could lose qualities that attract visitors and support unique ecosystems, Stephen Saunders, former deputy assistant secretary of the Interior Department, said in a report released by two advocacy groups."

Source: **AP**

Figure 14. An example of natural language data

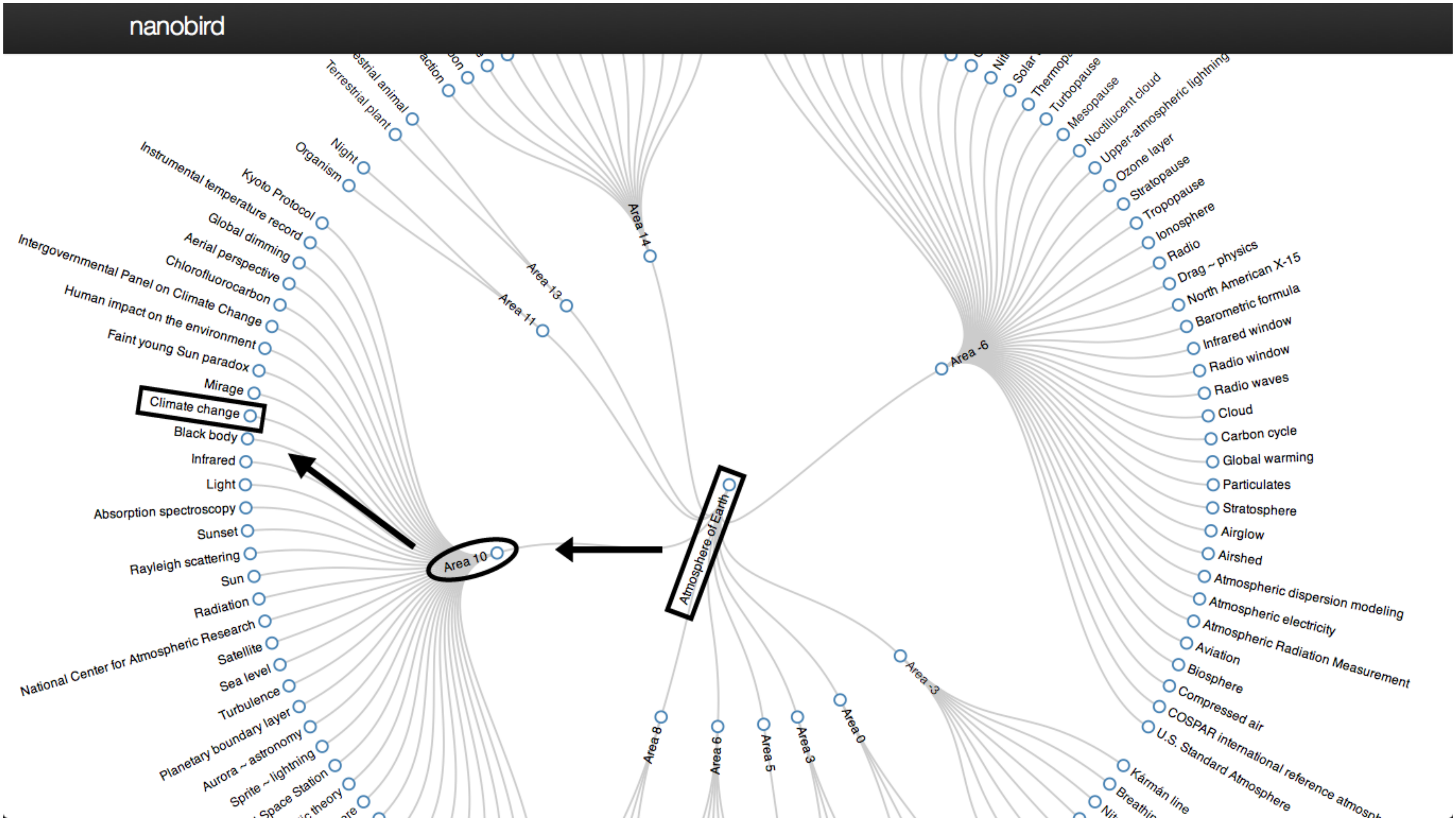


Figure 15. Partial fingerprint of the phrase "Atmosphere of Earth"

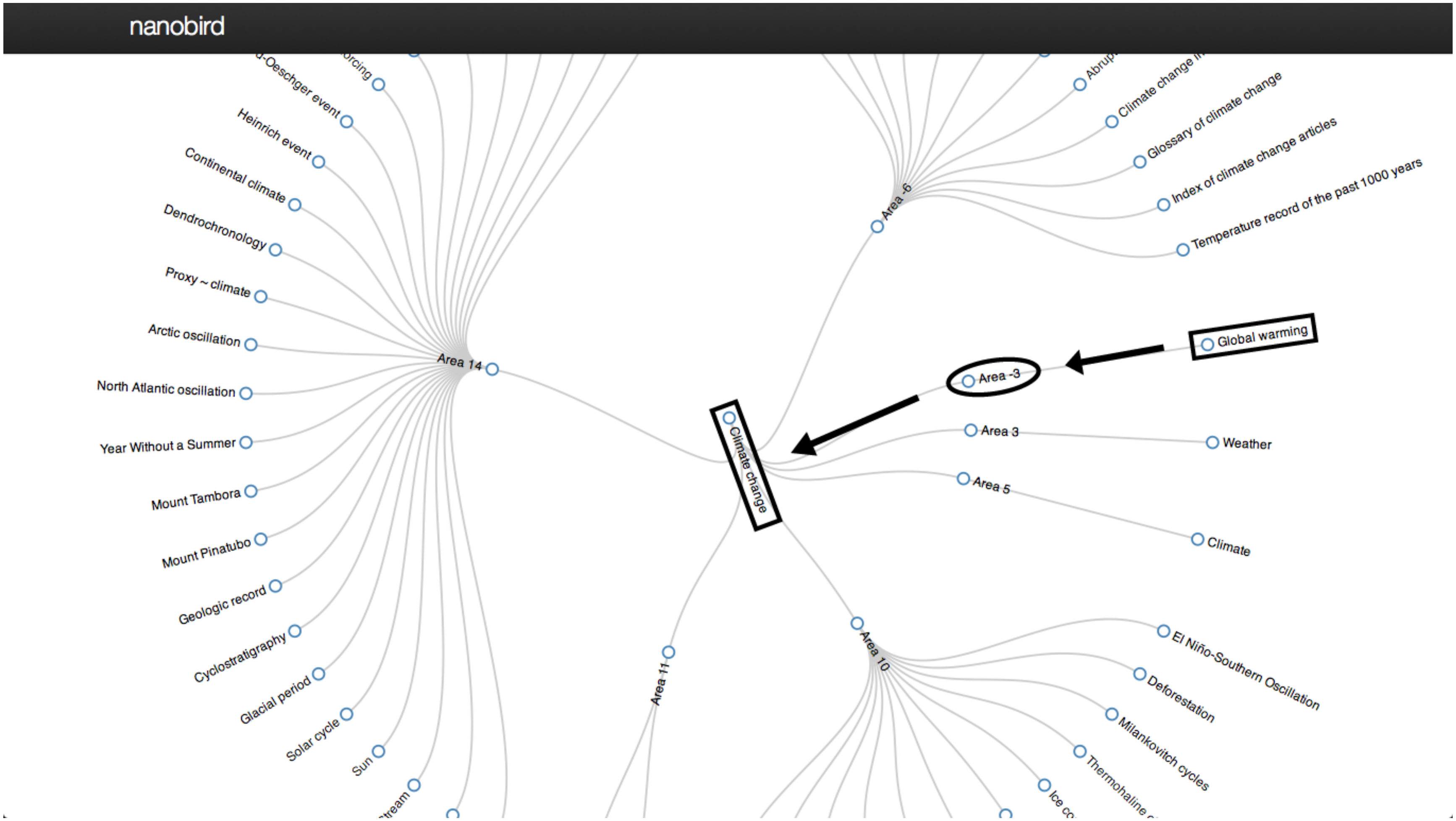


Figure 16. Partial fingerprint of the phrase "Climate change"

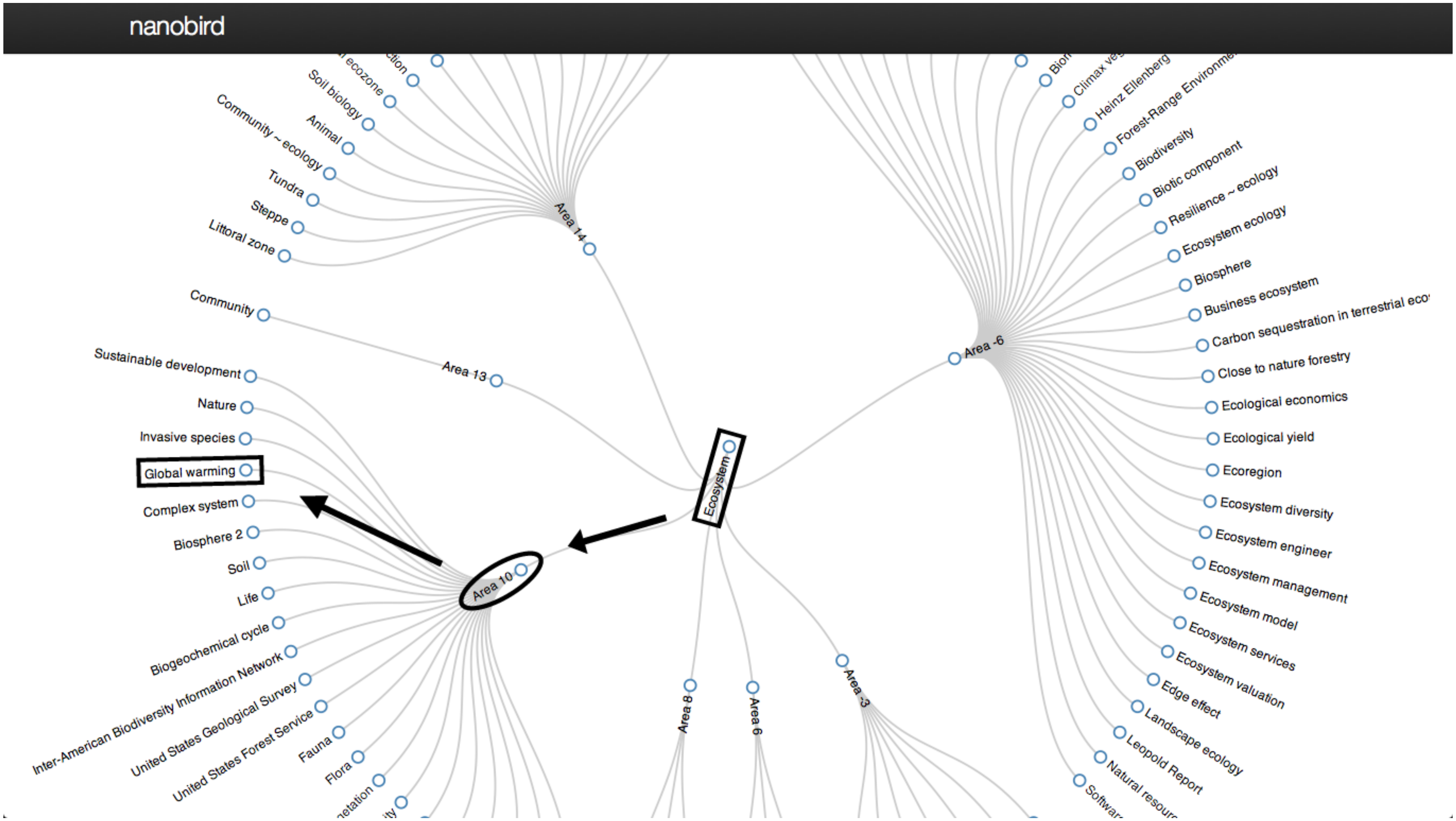


Figure 17. Partial fingerprint of the phrase "Ecosystem"

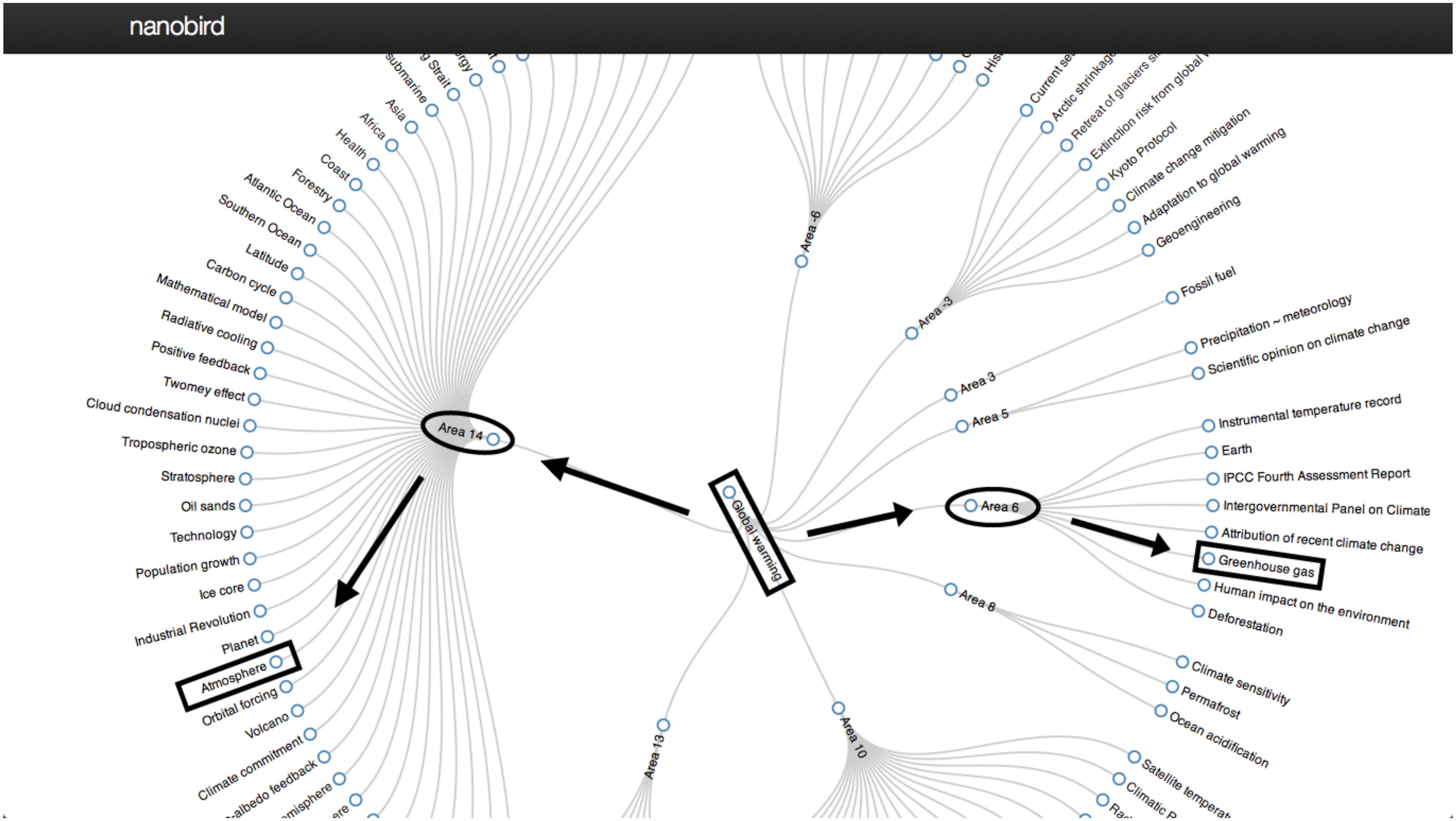


Figure 18. Partial fingerprint of the phrase "Global warming"

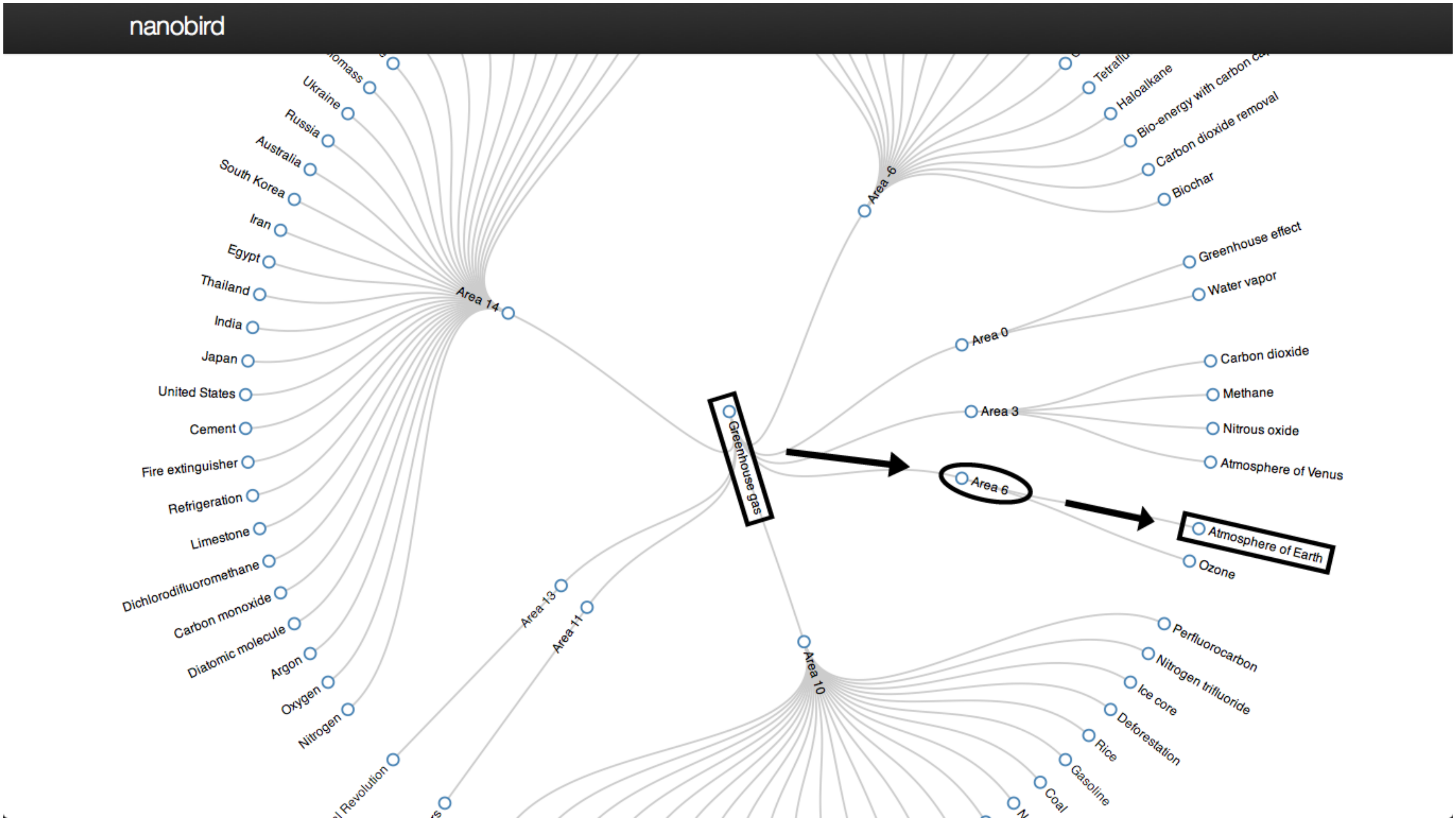


Figure 19. Partial fingerprint of the phrase "Greenhouse gas"

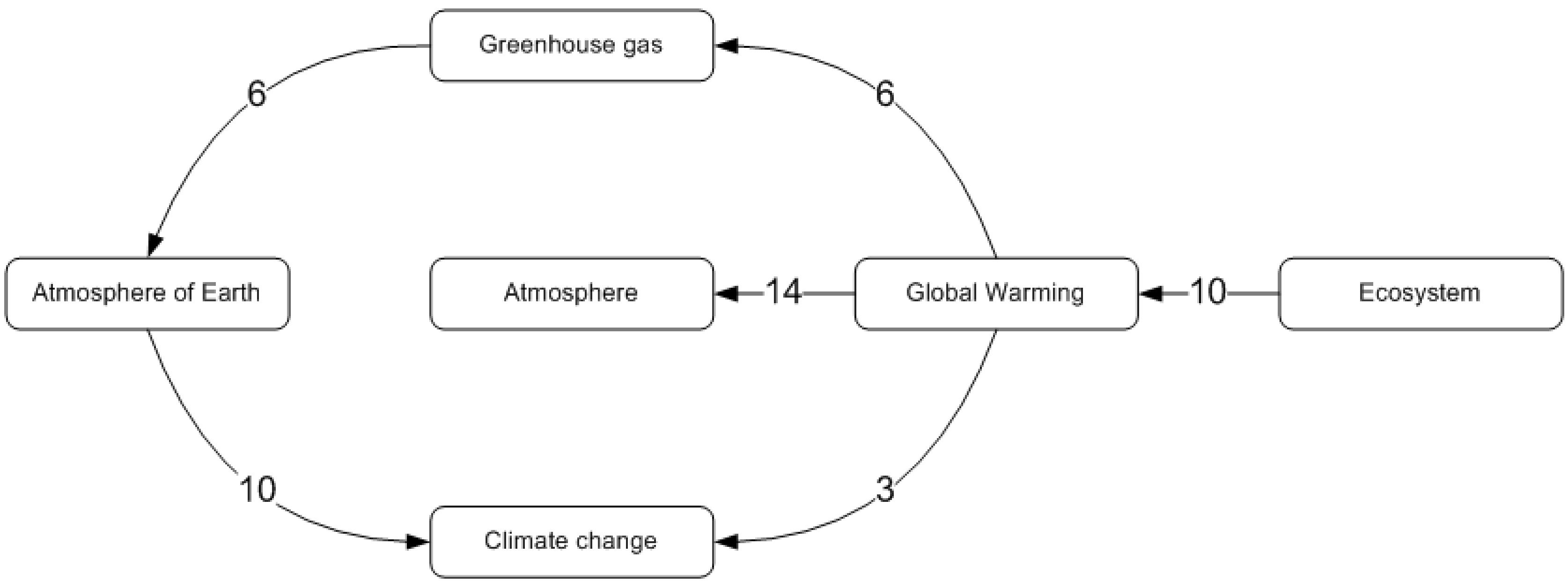


Figure 20. An example of a nanobird Context Analyzer Graph

@DrOz Interests*

Food
Health
Eating
Research
Nutrition
Physical exercise
Websites
Physical fitness

@NYTimeskrugman Interests*

Unemployment
Economy
Keynesian economics
Inflation
Monetary policy
Suffering
Interest rate
Macroeconomics
Fiscal policy
Economics
Middle East
Affect psychology
Affective science

@geostella Interests*

Sun
Solar energy
Energy
Renewable energy
Manufacturing
Technology
Solar power
Business

@AccelGrowth Interests*

Business
Investment
Technology
Venture capital
Companies

@billgates Interests*

Technology
Vaccination
Education
Investment
Funding
Teacher
World
Scientific method
Money
School
Science
Poverty

@MikeBloomberg Interests*

New York City
Michael Bloomberg
Mayor of New York City
Charitable organization
New York
Bloomberg L.P.
Philanthropy
Manhattan
Foundation non-profit
Health
Economy

Figure 21. Example showing the interest of some Twitter users after being processed by the twitter application using nRE