Categorial Dependency Grammars

Michael Dekhtyar [a,1] Alexander Dikovsky [b,1]

[a] *Dept. of CS, Tver St. Univ., Tver, Russia, 170000*
Michael.Dekhtyar@tversu.ru

[b] *LINA, Université de Nantes, 2, rue de la Houssinière, BP 92208 44322 Nantes Cedex 03 France*
Alexandre.Dikovsky@lina.univ-nantes.fr

Abstract

Categorial Dependency Grammars (CDG) introduced in this paper express projective and distant dependencies in classical categorial grammar terms. They treat order constraints in terms of oriented polarized valencies and a bounded commutativity rule. CDGs are expressive, constitute a convenient frame for coupling dependency grammars with linguistic semantics, and with all this, they are parsed in polynomial time under realistic conditions.

Key words: Dependencies, discontinuous constituents

---

1 Introduction

Dependency grammars (DGs) and categorial grammars (CGs) have much in common from the technical point of view. Both are completely lexicalized, use syntactic types in the place of rewriting rules, naturally fit functional semantic structures and are equivalent to CF-grammars if only the weak expressive power is concerned and the core syntax is considered. But as far as the matter concerns the strong expressive power, many fundamental differences appear between these formalisms. CGs are more adapted to syntagmatic (phrase) structures, whereas DGs are designed for assigning dependency trees. It is

---

[1] This work was sponsored by the Russian Fundamental Studies Foundation (Grant 01-01-00278).

true that there is a simple translation from phrase structures with head selection to projective [2] dependency trees and back (see [Gla66,Rob70] or [DM00] for more details), which perfectly conforms with the direct simulation of core dependency grammars by classical CGs [Gai61]. Unfortunately, this technical correspondence does not preserve the intended syntactic types. The reason is that the intended syntactic functions corresponding to the dependencies are different from those of the heads in the syntagmatic structures originating from the X-bar theory [Jac77]. Basically, the most essential distinctions are in the interpretation of verb and noun modifiers, which in dependency surface syntax are *subordinate* and *iterated*. On the other hand, the CGs' elimination rules induce dependencies from the functional type words to the argument type words. So, e.g. the adjectives, whose canonical type is $[n/n]$ must govern the modified nouns and not vice versa as in DGs. The same example illustrates the difference in treating iterated modifiers. In more recent DGs (cf. [ST93,LL96]) the explicitly used iterated dependencies preserve the subordinacy depth, whereas in CGs the simulation of the iteration through recursion leads to unlimited depth of subordinate modifiers. Even the subcategorization dependencies between verbs and their actants are quite different. They are more numerous in dependency syntax, in which dependencies reflect differences in pronominalization, redistribution and order constraints (see [Mel88] for more details). Another important difference is that dependency trees, in contrast with phrase structures, naturally capture discontinuous surface word order. Rather expressive (and so expensive) extensions of CGs are needed to cope with discontinuous structures and with naturally oriented dependencies simulation (e.g. non-associative and associative Lambek calculus and their multi-modal extensions [Mor94,MM]). Meanwhile, as it was shown in [Dik01], both can be very naturally and feasibly expressed in DGs in terms of *polarized dependency valencies* controlled by the simple principle, which enables a discontinuous dependency between two closest words having the same valency with the opposite signs ("first available" (**FA**) principle). A certain inconvenience of these polarized DGs is that being tree generating grammars, they are not completely lexicalized and do not propose a natural frame for formal linguistic semantics. This defect was eliminated in recent paper [Dik04], where the idea of polarized dependency valencies is implemented in terms of categorial grammars extended by the rule **FA**. In this paper we use a bounded commutativity rule in the place of the rule **FA**. The resulting *categorial dependency grammars* are enough expressive and universal to be used in practice, are parsed in polynomial time for each given inventory of dependency relations and can be naturally coupled with traditional type logical semantics via formal linguistic meaning as defined in [Dik03].

---

[2] *Projective* corresponds to *continuous* in terms of DTs: the projections of all words fill continuous segments of the sentence.

2 Syntactic types

We address to syntactic types as *categories*. Elementary categories are neutral. They serve as types of local dependencies. For instance, *subj* is the elementary local dependency, whose subordinate is a noun or a pronoun in the syntactic role of the subject and whose governor is a verb, whereas *subj_inf* is that in which the subordinate is a verb in infinitive. The set of elementary categories will be denoted by $\mathbf{C}$. Elementary categories may be *iterated*. For $a \in \mathbf{C}$, a^* denotes the corresponding *iterative* category. For instance, $modif^*$ is the type of iterated category $modif$. For a set $X \subseteq \mathbf{C}$, $X^* = \{C^* \mid C \in X\}$ and $X^\omega = X \cup X^*$. The iterative categories are also neutral.

Besides the neutral categories there are also polarized categories. They serve as types of distant dependencies. The polarized categories have one of four *polarities*: left and right positive $\nearrow, \nwarrow$ and left and right negative $\searrow, \swarrow$. For each polarity v, there is the unique "dual" polarity $\breve{v}$: $\breve{\nwarrow} = \swarrow$, $\breve{\swarrow} = \nwarrow$, $\breve{\nearrow} = \searrow$, $\breve{\searrow} = \nearrow$. Intuitively, the positive categories can be seen as valencies of the outgoing distant dependencies of governors, and the negative categories as those of the incoming distant dependencies of subordinate words. So they correspond respectively to the beginnings and the ends of distant dependencies. For instance, the positive valency $\nwarrow$ *wh_upon_obj* marks the beginning of the distant dependency *wh_upon_obj* of a transitive verb governing a left-dislocated object wh-group headed by the preposition 'UPON', whereas the end of this dependency is marked by the dual negative valency $\swarrow$ *wh_upon_obj* of this preposition (cf. *upon what dependency theory we rely*).

$\nearrow\mathbf{C}$, $\nwarrow\mathbf{C}$, $\searrow\mathbf{C}$ and $\swarrow\mathbf{C}$ denote the corresponding sets of polarized distant dependency categories. For instance, $\nearrow\mathbf{C} = \{(\nearrow C) \mid C \in \mathbf{C}\}$ is the set of *right positive* categories. $V^+(\mathbf{C}) = \nearrow\mathbf{C} \cup \nwarrow\mathbf{C}$ is the set of positive distant dependency categories, $V^-(\mathbf{C}) = \searrow\mathbf{C} \cup \swarrow\mathbf{C}$ is the set of those negative.

Defining distant dependencies, it is sometimes necessary to express that the subordinate word is the first (last) in the sentence, in the clause, etc., or it immediately precedes (follows) some word. For instance, in French the negative dependency category $\swarrow clit{-}dobj$ of a cliticized direct object must be anchored to the auxiliary verb or to the verb in a non-analytic form. For that we distinguish in the set of all negative distant categories a subset $Anc(\mathbf{C}) \subseteq V^-(\mathbf{C})$ of *anchored* negative categories.

Definition 1 *The set* $Cat(\mathbf{C})$ *of* dependency tree (DT) categories *is the least set verifying the conditions:*
1. $\mathbf{C} \cup V^-(\mathbf{C}) \subset Cat(\mathbf{C})$.
2. For $C \in Cat(\mathbf{C})$, $A_1 \in \mathbf{C}^\omega \cup \nwarrow\mathbf{C}$, $A_2 \in \mathbf{C}^\omega \cup \nearrow\mathbf{C}$, *and* $B \in Anc(\mathbf{C})$, *the categories* $[A_1\backslash C]$, $[C/A_2]$, $[B\backslash\backslash C]$ *and* $[C/\!/B]$ *also belong to* $Cat(\mathbf{C})$.

We suppose that all constructors $\backslash$, $/$, $\backslash\!\!\backslash$, $/\!\!/$ are associative. So every complex DT category α can be presented in the form

$$\alpha = [L_k\ l_k\ \ldots\ L_1\ l_1\ C\ r_1\ R_1\ \ldots\ r_m\ R_m],$$

where l_i and r_j are respectively left and right constructors. E.g.,

$$[(\swarrow clit{-}dobj)\backslash\!\!\backslash subj\backslash S/auxPP]$$

is one of categories of an auxiliary verb, which defines it as the host word for a cliticized direct object, requires the local subject dependency on its left and requires on its right the local dependency $auxPP$ with a subordinate PP.

3 Grammar definition

Definition 2 *A* categorial dependency grammar (CDG) *is a system* $G = (W, \mathbf{C}, S, \delta)$, *where* W *is a finite set of words,* $\mathbf{C}$ *is a finite set of elementary categories containing the selected* root category S, *and* δ *- called* lexicon *- is a finite substitution on* W *such that* $\delta(a) \subset Cat(\mathbf{C})$ *for each word* $a \in W$.

Below, we will index DT categories by their positions in a given sentence $w = a_1 \ldots a_n$. These indexes will serve to define dependency trees: α^i will be a category of a DT with the root a_i.

Definition 3 *A* D-sentential form *of a sentence* $w = a_1 \ldots a_n \in W^+$ *is a pair* (Δ, Γ), *where* Δ *is an oriented labelled graph with the set of nodes* $V = \{a_1, \ldots, a_n\}$ *and a set of arcs labeled by primitive categories, and* Γ *is a nonempty string of rooted categories.*
An initial D-sentential form *of* $w = a_1 \ldots a_n$ *is an expression* $((V, \emptyset), C_1^1 \ldots C_n^n)$, *in which* $C_i \in \delta(a_i)$ *for all* $1 \leq i \leq n$. *A* terminal D-sentential form *of* $w = a_1 \ldots a_n$ *is a pair* (Δ, S^j), *in which* $\Delta = (V, E)$ *is a DT on* w *with the root* a_j.

Below we define a provability relation $\vdash$. It is defined by several rules applying to sentential forms. The most specific are the rules of polarized dependency valencies' control. The idea behind these rules is that in order to establish a distant dependency between two words with dual dependency valencies, both valencies must be *charged*. Positive valencies are charged by definition. As to negative valencies, they may be charged or uncharged. The uncharged negative valencies can serve only to anchor a distant subordinate to a host word or position. As soon as the correct position of the subordinate is identified, its valency becomes charged and so available to the governor. In order to distinguish between charged and uncharged valencies, we use for each dependency valency vC its unique charged copy $\#(vC)$.

Definition 4 *Rules for provability relation* $\vdash$ *(we present only the rules* R^l *for left constructors; the right constructor rules* R^r *are similar).*

Simplification rule *:*

$\mathbf{S}$. $((V,E),\Gamma_1[C]^j\Gamma_2) \vdash ((V,E),\Gamma_1 C^j\Gamma_2)$ *for* $C \in \mathbf{C} \cup V^-(\mathbf{C})$.

Local dependency rule *:*

$\mathbf{L}^l$. $((V,E),\Gamma_1 C^j[C\backslash\beta]^l\Gamma_2) \vdash ((V,E\ \cup\{a_j \xleftarrow{C} a_l\}),\Gamma_1[\beta]^l\Gamma_2)$.

Iterative dependency rules*:*

$\mathbf{I}^\mathbf{l}$. $((V,E),\Gamma_1 C^j[C^*\backslash\alpha]^l\Gamma_2) \vdash ((V,E\ \cup\{a_j \xleftarrow{C} a_l\}),\Gamma_1[C^*\backslash\alpha]^l\Gamma_2)$.

$\mathbf{\Omega}^\mathbf{l}$. $((V,E),\Gamma_1[C^*\backslash\alpha]^l\Gamma_2) \vdash ((V,E),\Gamma_1\alpha^l\Gamma_2)$.

Anchored dependency rule:

$\mathbf{A}^\mathbf{l}$. $((V,E),\Gamma_1 C^j[C\backslash\!\!\backslash\alpha]^l\Gamma_2) \vdash ((V,E),\Gamma_1\#(C)^j\alpha^l\Gamma_2)$ *for* $C \in Anc(\mathbf{C})$.

Positive dependency rule:

$\mathbf{P}^\mathbf{l}$. $((V,E),\Gamma_1[(\nwarrow C)\backslash\alpha]^j\Gamma_2) \vdash ((V,E),\Gamma_1\#(\nwarrow C)^j\alpha^j\Gamma_2)$.

Commutativity rules *(both) :*

$\mathbf{C}^\mathbf{l}$. $((V,E),\Gamma_1\ (C')^j\#(vC)^l\ \Gamma_2) \vdash ((V,E),\Gamma_1\ \#(vC)^l(C')^j\ \Gamma_2)$ *if* (vC) *is a left dependency valency (i.e.* $(vC) \in \nwarrow\mathbf{C} \cup \searrow\mathbf{C}$*) and the category* C' *has neither occurrences of* (vC) *nor of* $(\breve{v}C)$.

$\mathbf{C}^\mathbf{r}$. $((V,E),\Gamma_1\ \#(vC)^j(C')^l\ \Gamma_2) \vdash ((V,E),\Gamma_1\ (C')^l\#(vC)^j\ \Gamma_2)$ *if* (vC) *is a right dependency valency (i.e.* $(vC) \in \nearrow\mathbf{C} \cup \swarrow\mathbf{C}$*) and the category* C' *has neither occurrences of* (vC) *nor of* $(\breve{v}C)$.

Distant dependency rule:

$\mathbf{D}^\mathbf{l}$. $((V,E),\Gamma_1\#(\swarrow C)^j\#(\nwarrow C)^l\Gamma_2) \vdash ((V,E\ \cup\{a_j \xleftarrow{C} a_l\}),\Gamma_1\Gamma_2)$.

This system of rules defines the immediate provability relation $\vdash$ *. Its subsystem consisting of simplification, local dependency and iterative category rules defines the* projective *immediate provability relation* $\vdash_p$. $\vdash^*$ *and* $\vdash_p^*$ *denote their corresponding reflexive-transitive closures.*

Definition 5 *A dependency tree (DT)* D *is assigned by a CDG* $G = (W, \mathbf{C}, S, \delta)$ *to a sentence* w *(denoted* $G(D,w)$*) if* $(\Delta_0,\Gamma_0) \vdash^* (D, S^j)$ *for some initial sentential form* (Δ_0,Γ_0) *of* w *and some* $1 \le j \le n$.
The DT language *generated by* G *is the set of DTs* $DT(G) = \{D \mid \exists w\ G(D,w)\}$.
The language *generated by* G *is the set of sentences* $L(G) = \{w \mid \exists D\ G(D,w)\}$.

This definition is correct in the following sense:

Proposition 1 *For each CDG G and sentence $w \in W^*$, if $(\Delta_0, \Gamma_0) \vdash^* (D, S^j)$ for some initial sentential form (Δ_0, Γ_0) of w, some $1 \leq j \leq n$, and some graph D, then D is a DT on w.*

4 Expressive power

CDGs are weakly more expressive than CFGs and strongly more expressive than dependency grammars generating projective DTs. This is shown by the following examples cited from [Dik04].

Example 1 *Let $G_0 = (\{a, b, c, d_1, d_2, d_3\}, C_0, S, \delta_0)$, where δ_0 is defined by:*

$a \mapsto [\swarrow C \backslash\backslash \swarrow B], [\swarrow B \backslash\backslash \swarrow B]$, $\quad d_1 \mapsto \swarrow C$,
$b \mapsto [\nwarrow B \backslash D / A]$, $\quad d_2 \mapsto [\swarrow B \backslash\backslash \nwarrow C \backslash S / D]$,
$c \mapsto [D \backslash A]$, $\quad d_3 \mapsto D$.

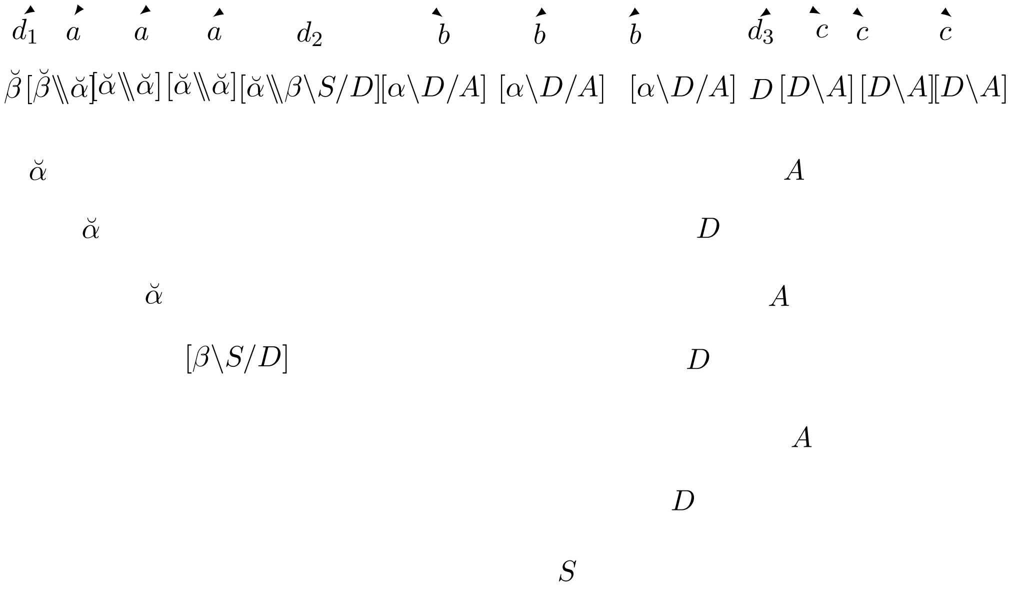

Fig. 1. A proof of $G_0(D, d_1 a^3 d_2 b^3 d_3 c^3)$.

Fig. 1 shows a proof of $G_0(D, d_1 a^3 d_2 b^3 d_3 c^3)$. In this proof, $\alpha = \nwarrow B$, $\beta = \nwarrow C$, two meeting continuous slanting lines correspond to one application of the local or iterated dependency rule, two meeting dashed slanting lines correspond to one application of the anchored dependency rule, and right-angled dashed lines connect categories to which the rules $\mathbf{P^l}, \mathbf{C^l}, \mathbf{C^r}, \mathbf{D^l}$ are applied.

Proposition 2 $L(G_0) = \{d_1 a^n d_2 b^n d_3 c^n \mid n > 0\}$.

Proof. In short, if we delete all polarized subcategories, we obtain a rigid categorial grammar defining the language $\{d_2 b^n d_3 c^n \mid n \geq 1\}$. Returning to the original CDG, we remark that: (1) only one occurrence of d_2 is possible (its functional subcategory is S), so exactly one occurrence of d_1 is possible and it must precede d_2, (2) if some a precedes d_1 or follows d_2, then its category cannot be eliminated, (3) only one a can have category $[\breve{\beta}\backslash\backslash\breve{\alpha}]$ because $\breve{\beta}$ assigned to d_1 becomes charged, (4) so there must be as many a as b. □

Example 2 *Fig. 2 shows a case of PP-movement in English expressed using the categories:*
$C_1 = [det\backslash subj/attr-rel] \in \delta(person)$,
$C_2 = [(\swarrow prepos)\backslash\backslash attr-rel/wh-rel] \in \delta(whom)$,
$C_3 = [subj\backslash wh-rel/inf-obj] \in \delta(must)$,
$C_4 = [(\nwarrow prepos)\backslash inf-obj] \in \delta(refer)$,
$C_5 = [subj\backslash S/n-copul] \in \delta(is)$,
$det \in \delta(the)$, $\swarrow prepos \in \delta(to)$, $subj \in \delta(you)$, *and* $n-copul \in \delta(Smith)$.

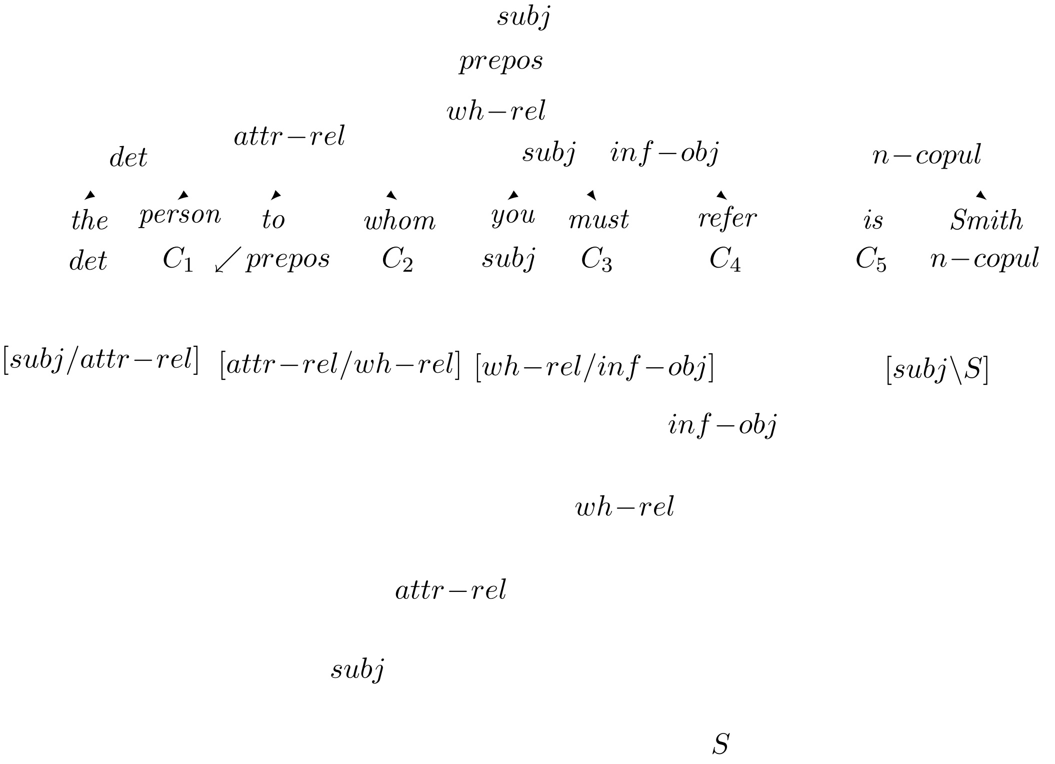

Fig. 2. A case of PP-movement in English.

In this proof, the rule $\mathbf{L^1}$ applied to det and C_1 gives the dependency ($the \xleftarrow{det} person$) of category $[subj/attr-rel]$. $\mathbf{L^1}$ applied to $subj$ and C_3 gives the dependency ($you \xleftarrow{subj} must$) of category $[wh-rel/inf-obj]$. $\mathbf{A^1}$ applied to $\swarrow prepos$ and C_2 gives $\#(\swarrow prepos)[attr-rel/wh-rel]$. Now we can eliminate $\#(\swarrow prepos)$ applying rules $\mathbf{C^l}, \mathbf{C^r}, \mathbf{P^l}, \mathbf{D^l}$ to this category and C_4. This reduces C_4 to $inf-obj$ and gives the distant dependency $(to \overset{wh-rel}{\longleftarrow} refer)$. Finally, $\mathbf{L^l}, \mathbf{L^r}$ are applied four times.

CDG can be strongly simulated by the polarized dependency tree grammars (PDTG) introduced in [Dik01].

Proposition 3 *For each CDG G_1 there is a PDTG G_2 such that for all $w \in W^+$ and all DTs D $G_1(D,w)$ iff $G_2(D,w)$.*

Proof. We illustrate the idea of this simulation by the following example. Suppose that $a \mapsto [A\backslash(\searrow B)\backslash\!\!\backslash(\nwarrow D)\backslash C/E^*]$ in G_1. Then in G_2 we will have the following tree rewriting rules:

$$N(C) \quad \overset{E}{} \overset{\blacktriangle}{N(E)} \parallel N(C_1) \quad \overset{E}{} \overset{\blacktriangle}{N(E)} \quad \to \quad N(C)$$

$$\overset{A}{} \overset{\blacktriangle}{N(A)} \quad N(C_2) \quad \to \quad N(C_1)$$

$$\overset{B}{--\!>} \quad N(B) \quad N(C_3) \quad \to \quad N(C_2)$$

$$\overset{D}{<\!--} \quad a \to \quad N(C_3)$$

It is not difficult to see that this construction gives a PDTG G_2 strongly equivalent to G_1. □

Definition 6 *Let D be a DT of a sentence $w = a_1 \ldots a_n$. For a space i between the words a_i and a_{i+1}, $1 \leq i < n$, we define the* distant dependencies thickness *in i (denoted $dth(D,i)$) as the number of distant dependencies $(a_k \overset{d}{<\!--} a_l)$, $(a_k \overset{d}{--\!>} a_l)$ in D covering i (i.e. such that $k < i < l$ for some k, l and d). $dth(D) =_{df} max\{dth(D,i) | 1 \leq i < |D|\}$ and $dth(G) =_{df} max\{0, min\{dth(D) | G(D,w)\} \mid w \in L(G)\}$.*

For instance, $dth(G_0) = \infty$. For natural languages, this measure is seemingly bounded by a small constant (2 or 3). In example 2 $dth(D) = 1$.

Theorem 1 *If for a CDG G, the measure $dth(G)$ is bounded by a constant, then $L(G)$ is context-free.*

Proof. For all PDTG G_1, $dth(G_1) \geq \sigma(G_1)$, where $\sigma(G_1)$ - the *defect* of G_1 - is a complexity measure of PDTGs defined in [Dik01]. So if a CDG G has a distant dependencies thickness bounded by a constant k, then the PDTG G' simulating G has a defect bounded by the same constant. Therefore, according to Theorem 1 in [Dik01], $L(G') = L(G)$ is context-free. □

It is an interesting theoretical problem to compare the weak generative capacity of CDGs and mildly context-sensitive grammars [JSW91]. We conjecture that the 2-copy language $\{wcw \mid w \in W^*\}$ cannot be generated by CDGs. On the other hand, the following proposition shows that CDG-languages are incomparable with basic TAG languages.

Proposition 4 *Each language* $L^{(m)} = \{d_0 a_0^n d_1 a_1^n \ldots d_m a_m^n d_{m+1} | n \geq 0\}$ *is generated by a CDG.*

Proof. An argument similar to that in Proposition 2 shows that $L^{(m)}$ is generated by the following CDG:
$d_0 \mapsto [S/D_0]$, $d_{m+1} \mapsto D_m$, and $a_0 \mapsto [D_0/D_0/(\nearrow A_m)/\ldots/(\nearrow A_1)]$,
$d_i \mapsto [D_{i-1}/\!\!/(\searrow A_i)]$, and $a_i \mapsto [(\searrow A_i)/\!\!/(\searrow A_i)], [(\searrow A_i)/D_i]$ for all $0 < i \leq m$. □

5 Complexity

If there is no uniform constraint on the number of elementary categories, then parsing of CDGs is a hard problem.

Theorem 2 *The problem* $G(D, w)$ *is NP-complete.*

Proof. Its NP-hardness can be proven by the following polynomial reduction of $3-CNF$. Let $\Phi = C_1 \wedge \ldots \wedge C_m$ be a CNF over variables $x_1, ..., x_n$, in which clauses C_j include 3 literals l_1^j, l_2^j, l_3^j and $l_k^j \in \{x_1, \neg x_1, \ldots, x_n, \neg x_n\}$.
Let $G(\Phi) = (W, \mathbf{C}, S, \delta)$, where $W = \{\Phi, C_1, \ldots, C_m, x_1, \ldots, x_n, y_1, \ldots y_n\}$, $\mathbf{C} = \{S, A, 1_0, 1_1, 2_0, 2_1, \ldots, n_0, n_1\}$ and
$\delta(\Phi) = [(A\backslash)^n\backslash S]$, $\delta(x_i) = \{[A/(\nearrow i_0)], [A/(\nearrow i_1)]\}$, $\delta(y_i) = \{(\searrow i_0), (\searrow i_1)\}$, $\delta(C_j) = \{cat(l_1^j), cat(l_2^j), cat(l_3^j)\}$, where $cat(x_i) = [(\searrow i_1)/(\nearrow i_1)]$ and $cat(\neg x_i) = [(\searrow i_0)/(\nearrow i_0)]$. Let also $w(\Phi) = x_1 x_2 \ldots x_n \Phi C_1 C_2 \ldots C_m y_1 y_2 \ldots y_n$.
Assertion. Φ is satisfiable iff $(\exists D : DT)\ G(\Phi)(D, w(\Phi))$. □

Fortunately, this anomaly is unrealistic: for each language the inventory of elementary categories is fixed. With such a uniform bound CDG parsing has polynomial complexity. It turns out that to parse a CDGs, it suffice to perform two *independent* tests: the first in terms of the projective provability $\vdash_p$ and the second in terms of neutralizability of distant dependency valencies. To formulate this fact, we need two different projections of categories. The first, called *local*, preserves only elementary and anchored argument sub-categories. Intuitively, it preserves only projective dependencies of words and also their neighborhood of anchored words. The second projection, called *valency pro-*

jection, preserves only polarized sub-categories and their respective order.

Definition 7 Local projection $\|\gamma\|_l$ *of a string* $\gamma \in Cat(\mathbf{C})^*$ *is defined as follows:*

l1. $\|\varepsilon\|_l = \varepsilon$; $\|C\gamma\|_l = \|C\|_l\|\gamma\|_l$ *for* $C \in Cat(\mathbf{C})$ *and* $\gamma \in Cat(\mathbf{C})^*$.

l2. $\|C\|_l = C$ *for* $C \in \mathbf{C}^\omega \cup Anc(\mathbf{C})$.

l3. $\|C\|_l = \varepsilon$ *for loose distant dependency categories* C.

l4. $\|[a\backslash\alpha]\|_l = [a\backslash\ \|\alpha\|_l]$ *and* $\|[\alpha/a]\|_l = [\|\alpha\|_l/a]$ *for* $a \in \mathbf{C}^\omega$ *and* $\alpha \in Cat(\mathbf{C})$.

l5. $\|[\nwarrow a\backslash\alpha]\|_l = \|[\alpha/ \nearrow a]\|_l = \|\alpha\|_l$ *for all* $a \in \mathbf{C}$ *and* $\alpha \in Cat(\mathbf{C})$.

l6. $\|[\alpha_1\backslash\!\backslash\alpha_2]\|_l = [\alpha_1\backslash\|\alpha_2\|_l]$ *and* $\|[\alpha_2/\!/\alpha_1]\|_l = [\|\alpha_2\|_l/\alpha_1]$ *for anchored dependencies* $\alpha_1 \in Anc(\mathbf{C})$ *and* $\alpha_2 \in Cat(\mathbf{C})$.

Valency projection $\|\gamma\|_v$ *of a string* $\gamma \in Cat(\mathbf{C})^*$ *is defined as follows:*

v1. $\|\varepsilon\|_v = \varepsilon$; $\|C\gamma\|_v = \|C\|_v\|\gamma\|_v$ *for* $C \in Cat(\mathbf{C})$ *and* $\gamma \in Cat(\mathbf{C})^*$.

v2. $\|C\|_v = \varepsilon$ *for* $C \in \mathbf{C}^\omega$.

v3. $\|C\|_v = C$ *for* $C \in V(\mathbf{C})$.

v4. $\|[\alpha]\|_v = \|\alpha\|_v$ *for all* $[\alpha] \in Cat(\mathbf{C})$.

v5. $\|A\backslash\!\backslash\alpha\|_v = \|\alpha/\!/A\|_v = \|\alpha\|_v$.

v6. $\|\alpha_1\backslash\alpha_2\|_v = \|\alpha_1\backslash\!\backslash\alpha_2\|_v = \|\alpha_1/\alpha_2\|_v = \|\alpha_1/\!/\alpha_2\|_v = \|\alpha_1\|_v\|\alpha_2\|_v$.
for all $\alpha_1, \alpha_2 \in Cat(\mathbf{C})$.

Example 3 *According to these definitions,*

$$\|[(\searrow c)\backslash\!\backslash(\nwarrow a)\backslash b\backslash(\swarrow d)/e]\|_l \;=\; \begin{cases} [(\searrow c)\backslash b\backslash(\swarrow d)/e], & if\ \swarrow d \in Anc(\mathbf{C}), \\ [(\searrow c)\backslash b\backslash\varepsilon/e], & otherwise \end{cases}$$

and $\|[(\searrow c)\backslash\!\backslash(\nwarrow a)\backslash b\backslash d]\|_v \;= \nwarrow a$, $\|[(\searrow c)\backslash\!\backslash(\nwarrow a)\backslash b\backslash(\swarrow d)/e]\|_v \;= \nwarrow a \swarrow d$ *independent of which is* $\swarrow d$ *: anchored or loose.*

Slightly abusing notation, below we will apply the relation $\vdash_p$ to local projections of D-sentential forms. These forms may contain occurrences of anchored categories. In the local projection, these anchored categories are treated just as elementary categories.

Besides these projections we need a criterion of "well-bracketed" polarized categories. In this bracketing, $\swarrow d$ and $\nearrow d$ play the role of left brackets and $\nwarrow d$ and $\searrow d$ serve as the corresponding right brackets. Obviously, for a left valency α, the corresponding right valency is $\breve{\alpha}$. We will call the pair $(\alpha, \breve{\alpha})$ *correct.*

Definition 8 *Let* $G = (W, \mathbf{C}, S, \delta)$ *be a CDG,* $(\alpha, \breve{\alpha})$ *be a correct pair and* $\gamma \in Cat(\mathbf{C})^+$ *be a string of categories. For both dependency valencies* β *in* $(\alpha, \breve{\alpha})$, $|\gamma|_\beta$ *will denote the number of occurrences of* β *in the valency projection* $\|\gamma\|_v$.

The values

$$\Delta^L_\alpha(\gamma) = max\{|\gamma'|_{\breve{\alpha}} - |\gamma'|_\alpha \mid \gamma' \textit{ is a } \textsf{prefix} \textit{ of } \gamma\},$$
$$\Delta^R_\alpha(\gamma) = max\{|\gamma'|_\alpha - |\gamma'|_{\breve{\alpha}} \mid \gamma' \textit{ is a } \textsf{suffix} \textit{ of } \gamma\}$$

express respectively the number of right and left non-neutralized dependency valencies α (i.e. the maximal deficit of left and right α−parentheses) in γ [3].

Let $\gamma, \gamma_1, \gamma_2 \in Cat(\mathbf{C})^+$ be some strings of categories and $(\alpha, \breve{\alpha})$ be a correct valency pair.
1. If $|\gamma|_\alpha = |\gamma|_{\breve{\alpha}} = 0$, then the pair $(\alpha, \breve{\alpha})$ is neutralized *in γ.*
2. If $(\alpha, \breve{\alpha})$ is neutralized *in γ, γ_1, and γ_2, then it is also neutralized in $\gamma_1\alpha\gamma\breve{\alpha}\gamma_2$.*

Finally, the use of iterative types leads to the following notion of realization.

Definition 9 *For a category $C = [\alpha D^*\backslash\beta]$, the categories $[\alpha\beta]$, $[\alpha D\backslash\beta]$, $[\alpha D\backslash D\backslash\beta]$, $[\alpha D\backslash D\backslash D\backslash\beta]$, etc. are* realizations *of C (similar for right iterative categories). Replacing in a string of categories $\gamma \in Cat(\mathbf{C})^+$ each category having iterative subcategories by some its realization we obtain a realization of γ. Let $R(\gamma)$ denote the set of all realizations of γ.*

Here is the two-test membership criterion.

Theorem 3 *Let $G = (W, \mathbf{C}, S, \delta)$ be a CDG. $x \in L(G)$ iff there exist a string of categories $\alpha \in \delta(x)$ and some its realization $\gamma \in R(\alpha)$ such that:*
1. $\|\gamma\|_l \vdash^*_p S$,
2. each correct pair $(\alpha, \breve{\alpha})$ is neutralized in $\|\gamma\|_v$.

Proof. Its main part is the proof of the following lemma.

Lemma 1 *The system of rules defining $\vdash$ is equivalent to the system in [Dik04], in which in the place of the rules $\mathbf{P^l}, \mathbf{P^l}, \mathbf{C^l}, \mathbf{C^r}$ and $\mathbf{D^l}, \mathbf{D^r}$ there are the following distant dependency rules $\mathbf{D^l_{FA}}, \mathbf{D^r_{FA}}$:*
$\mathbf{D^l_{FA}}$. $\Gamma_1\#(\swarrow C)\Gamma_2[(\nwarrow C)\backslash\alpha]\Gamma_3 \vdash \Gamma_1\Gamma_2\alpha\Gamma_3$.
The rule applies if there are no occurrences of categories $\swarrow C$ and $\nwarrow C$ in Γ_2.

This theorem enables efficient parsing algorithms for CDGs.

Algorithm pars we describe below is Earley style [Ear70]. When applied to a string $x = w_1...w_n$, **pars** incrementally fills a triangular matrix M of size $n \times n$, whose element $M[i, j], i < j$, is a finite set of so called "*items*" [4].

For a given CDG $G = (W, \mathbf{C}, S, \delta)$, let $L = (\alpha_1, \ldots, \alpha_p)$ be the list of all positive and negative left dependency valencies in $\nearrow\mathbf{C} \cup \swarrow\mathbf{C}$ and $R = (\breve{\alpha}_1, \ldots, \breve{\alpha}_p)$ be the list of all corresponding negative and positive right dependency valencies in $\searrow\mathbf{C} \cup \nwarrow\mathbf{C}$. Items of **pars** have the form (I, lc, rc), $lc = (\Delta^L_{\alpha_1}, \ldots, \Delta^L_{\alpha_p})$ and $rc = (\Delta^R_{\alpha_1}, \ldots, \Delta^R_{\alpha_p})$

---

[3] Having in mind that there is $\gamma' = \varepsilon$, the values $\Delta^L_\alpha(\gamma)$ and $\Delta^R_\alpha(\gamma)$ are non-negative.
[4] The lines are indexed from 0 to $n - 1$ and the columns from 1 to n.

are integer vectors, whose components are the corresponding differences of left and right dependency valencies, and $I = ({}_{\boxed{i}}[\,\alpha'{}_{\boxed{j}}D\backslash\alpha''\backslash D'/\beta\,]^j,\ lc, rc)$, or $I = ({}_{\boxed{i}}[\,\alpha\backslash D'/\beta''/D{}_{\boxed{j}}\beta'\,]^r,\ lc, rc)$, or at last, $I = ({}_{\boxed{i}}[\,\alpha\backslash D/\beta\,]^r{}_{\boxed{j}},\ lc, rc)$.
I of the first form represents categories with non-eliminated left-argument subtypes, the second, with eliminated left- and non-eliminated right-argument subtypes, and the third, with all subtypes eliminated.

pars uses operators **PROPOSE**, **SUBORDINATE_L** and **SUBORDINATE_R**. The last two are similar, so we show one of them.

In algorithm **PROPOSE**, (α_m) and $(\breve{\alpha}_m)$ denote the corresponding members of the lists L and R.

```
PROPOSE($i$) $(1 \le i \le n)$
FORALL $C = \|\gamma\|_l$ WHERE $\gamma \in \delta(w_i)$
  DO
        IF $C = [\,\alpha\backslash D/\beta\,]$ and $\alpha = D'\backslash\alpha'$ THEN
              $I = {}_{\boxed{i-1}}[\,{}_{\boxed{i}}D'\backslash\alpha'\backslash D/\beta\,]^i$
        ELSE  $(\alpha = \varepsilon)$
              $I = {}_{\boxed{i-1}}[\,\backslash D/\beta'/D'{}_{\boxed{i}}\,]^i$
        END_IF;
        FORALL $\alpha_i \in L$
        DO
              $(lc)_i = \Delta^L_{\alpha_i}(\gamma)$;  $(rc)_i = \Delta^R_{\alpha_i}(\gamma)$
        END_FORALL;
       add $(I, lc, rc)$ to $M[i-1, i]$
  END_FORALL

SUBORDINATE_L($i, j, k$) $(0 \le i < j < k \le n)$
FORALL  $({}_{\boxed{i}}[\,\alpha_1\backslash D_1/\beta_1\,]^r{}_{\boxed{j}}, lc_1, rc_1) \in M[i, j]$,  and
        $({}_{\boxed{j}}[\,\alpha'_2{}_{\boxed{k}}\alpha''_2 D'/\beta\,]^k, lc_2, rc_2) \in M[j, k]$
  DO
     FORALL $m$ WHERE $1 \le m \le p$ (in $lc$ and $rc$)
     DO
        $(lc)_m = (lc_1)_m + max\{0, (lc_2)_m - (rc_1)_m\}$;
        $(rc)_m = (rc_2)_m + max\{0, (rc_1)_m - (lc_2)_m\}$
     END_FORALL;
     FORALL left-argument-iterative $\omega_1, \omega_2$
        WHERE $\alpha''_2 = \omega_1 D_2\backslash\omega_2\gamma$ and $(D_2 = D_1$ or $D_2 = D_1^*)$
     DO
        IF  $\gamma \ne \varepsilon$ THEN
           add  $({}_{\boxed{i}}[\,\alpha'_2\omega_1 D_1\backslash\omega_2{}_{\boxed{k}}\gamma D'/\beta\,]^k, lc, rc)$  to $M[i, k]$
```

```
        ELSE_IF $\gamma = \varepsilon$ and $\beta \neq \varepsilon$ THEN
            add $(\boxed{i}\,[\,\alpha'_2\alpha''_2 D'/\beta\boxed{k}\,]^k, lc, rc)$ to $M[i,k]$
        ELSE $(\gamma = \varepsilon = \beta)$
            add $(\boxed{i}\,[\,\alpha'_2\alpha''_2 D'/\,]^k\boxed{k}, lc, rc)$ to $M[i,k]$
        END_IF;
        IF $D_2 = D_1^*$ THEN
            add $(\boxed{i}\,[\,\alpha'_2\omega_1\boxed{k}D_1^*\backslash\omega_2\gamma D'/\beta\,]^k, lc, rc)$ to $M[i,k]$
        END_IF;
        IF $D \notin Anc(\mathbf{C})$ THEN
            $\Gamma = \Gamma \cup \{r \xleftarrow{D} k\}$
        END_IF
    END_FORALL
END_FORALL
```

Algorithm pars
Input: $G \in \mathfrak{C}^{pD}$ and $x = w_1 ... w_n$
Output: A DT T on x.

```
FORALL $i$ incr $1 \leq i \leq n$
DO
    PROPOSE(i)
END_FORALL;
FORALL $k$ incr $1 \leq k \leq n$
DO
    FORALL $j$ decr $0 \leq j < k$
    DO
        FORALL $i$ incr $0 \leq i < j$:
        DO
            SUBORDINATE_L$(i, j, k)$
        END_FORALL
    END_FORALL;
    FORALL $j$ decr $0 \leq j < k$
    DO
        FORALL $i$ incr $0 \leq i < j$:
        DO
            SUBORDINATE_R$(i, j, k)$
        END_FORALL
    END_FORALL
END_FORALL
succeed when $(\boxed{0}\,[\,\alpha\backslash S/\beta\,]^r\boxed{n}, \bar{0}, \bar{0}) \in M[0,n]$ for some $\alpha, \beta, r$;
trace back the precursors of this item to identify the DT dependencies
```

pars is a correct and complete polynomial time parser of CDGs.

Theorem 4 **pars**(G, x) *succeeds iff* $x \in L(G)$.

Theorem 5 *Let* p *be the number of all polarized categories. Then:*
(1) **pars** *has time complexity* $\mathbf{O}(\mathsf{n}^{2\mathsf{p}+4})$,
(2) If p *and* $dth(G)$ *are constant, then* **pars** *has time complexity* $\mathbf{O}(n^4)$.

Remark. 1. In fact, the complexity of the membership problem $w \in L(G)$ is one order lower (there is no need in root indexes in the items): for instance, in case (2) of this theorem we would have $\mathbf{O}(n^3)$.
2. Clearly, if CDG G is *projective*, i.e. it doesn't use polarized dependency valencies, then $dth(G) = 0$ and the root indexes are also not needed.

Corollary 1 *If* p *is constant, then:*
(1) Projective CDGs are parsed in time $\mathbf{O}(n^3)$.
(2) If $dth(G)(n) = \mathbf{O}(\log n)$, *then* **pars** *has time complexity* $\mathbf{O}(n^{5+\varepsilon})$.

6 Acknowledgments

The authors are grateful to Denis Béchet and Annie Foret for helpful comments and fruitful discussions of this work.

7 Concluding remarks

The Categorial Dependency Grammars introduced in this paper combine type-driven style fitting well the standard methods of constructing formal semantics with valency/polarity style proper to dependency grammars. They can be easily adapted to practical definitions of surface dependency syntax of natural languages. For this, elementary types should be provided with nonrecursive feature structures and feature unification and propagation through dependencies must be allowed. The use of anchored categories and oriented polarities makes possible to express a variety of linear order constraints formulated in terms of the so called topological domains (cf.[Br8]). At the same time, the CDGs have the most efficient parsing algorithms as compared to other dependency grammars expressing unlimited distant dependencies.

References

[Br8] Norbert Bröker. Separating surface order and syntactic relations in a dependency grammar. In *Proc. COLING-ACL*, pages 174–180, Montreal, 1998.

[Dik01] Alexander Dikovsky. Polarized non-projective dependency grammars. In Ph. de Groote, G. Morill, and Ch. Retoré, editors, *Proc. of the Fourth Intern. Conf. on Logical Aspects of Computational Linguistics*, Lecture Notes in Artificial Intelligence. vol. 2099, pages 139–157. Springer, 2001.

[Dik03] Alexander Dikovsky. Linguistic meaning from the language acquisition perspective. In G. Jäger, P. Monachesi, G. Penn, and S. Winter, editors, *Proc. of the 8th Intern. Conf. "Formal Grammar 2003" FG 2003*, Vienna, Austria, 2003. Vienna Techn. Univ.

[Dik04] Alexander Dikovsky. Dependencies as categories. 2004. Submitted.

[DM00] Alexander Dikovsky and Larissa Modina. Dependencies on the other side of the curtain. *Traitement Automatique des Langues (TAL)*, 41(1):79–111, 2000.

[Ear70] Jay Earley. An efficient context-free parsing algorithm. *Communications of the ACM*, 13:94–102, 1970.

[Gai61] Haïm Gaifman. Dependency systems and phrase structure systems. Report p-2315, RAND Corp. Santa Monica (CA), 1961. Published in: Information and Control, 1965, v. 8, n 3, pp. 304-337.

[Gla66] Alexej V. Gladkij. *Lekcii po Matematičeskoj Lingvistike dlja Studentov NGU [Course of Mathematical Linguistics. Novosibirsk State University (Russ.)]. (French transl. Leçons de linguistique mathématique. facs. 1, 1970, Dunod).* Novosibirsk State University, 1966.

[Jac77] Ray Jackendoff. *X-bar Syntax. A Study of Phrase Structure.* MIT Press, Cambridge, 1977.

[JSW91] Aravind K. Joshi, Vijay K. Shanker, and David J. Weir. The convergence of mildly context-sensitive grammar formalisms. In P. Sells, S. Shieber, and T. Wasow, editors, *Foundational issues in natural language processing*, pages 31–81, Cambridge, MA, 1991. MIT Press.

[LL96] Vincenzo Lombardo and Leonardo Lesmo. An earley-type recognizer for dependency grammar. In *Proc. 16th COLING*, pages 723–728, 1996.

[Mel88] Igor Mel'čuk. *Dependency Syntax.* SUNY Press, Albany, NY, 1988.

[MM] Michael Moortgat and Glin V. Morrill. Heads and phrases. Type calculus for dependency and constituent structure. Ms OTS, Utrecht.

[Mor94] Glin V. Morrill. *Type Logical Grammar. Categorial Logic of Signs.* Kluwer Academic Publishers, Dordrecht, 1994.

[Rob70] Jane J. Robinson. Dependency structures and transformational rules. 46(2):259–285, 1970.

[ST93] Daniel Sleator and Davy Temperly. Parsing English with a Link Grammar. In *Proc. IWPT'93*, pages 277–291, 1993.