DIMACS Technical Report 97-21
June 1997

Bounded Hyperset Theory and Web-like Data Bases [1]

by

Alexei Lisitsa[2,3] Vladimir Sazonov[4,5]

[1]This paper is accompanying to DIMACS TR-97-05. To be presented in "Kurt Gödel Colloquium", Vienna, 25–29 August, 1997, and in the Workshop "Logic and Algorithms—One Year Later", DIMACS, Rutgers University, 23–25 July, 1997

[2]Permanent Member: Program Systems Institute of Russian Acad. of Sci., Pereslavl-Zalessky, 152140, Russia. e-mail: `alexei@lisitsa.botik.ru`

[3]Supported by RBRF (project 96-01-01717) and by INTAS (project 93-0972).

[4]Permanent Member: Program Systems Institute of Russian Acad. of Sci., Pereslavl-Zalessky, 152140, Russia. e-mail: `sazonov@logic.botik.ru`

[5]Supported by RBRF (project 96-01-01717), by INTAS (project 93-0972) and Swedish Royal Academy of Sciences.

DIMACS is a partnership of Rutgers University, Princeton University, AT&T Labs, Bellcore, and Bell Labs.

DIMACS is an NSF Science and Technology Center, funded under contract STC–91–19999; and also receives support from the New Jersey Commission on Science and Technology.

ABSTRACT

It is demonstrated how a Hyperset Theory (satisfying P.Aczel's Anti-Foundation Axiom) naturally arises from the idea of World-Wide Web (WWW). Alternatively, Web serves as an illustration and possible application of the abstract notion of antifounded sets. A Δ-language of Bounded Hyperset Theory is presented as a query language to the Web or, more generally, to Web-like Data Bases (WDB). It is shown that it describes exactly all abstract (or generic, up to bisimulation) PTIME-computable queries with respect to (possibly cyclic) graph encoding of hypersets. This result is based (i) on reducing of the Δ-language for hypersets to the language FO+LFP (= First-Order Logic with the Least Fixed-Point operator) over graphs considered up to bisimulation relation and (ii) on definability in FO+LFP of a linear ordering on any strongly extensional finite graph by a single formula (cf. also DIMACS TR-97-05). The case of finitely branching, possibly infinite graphs and corresponding hypersets is also discussed. It corresponds to finitely-branching, but infinite Web. However, it deserves special further investigation.

1 Introduction

We present in this paper rather abstract, "static" set-theoretic view on the World-Wide Web (WWW) or, more generally, on Web-*like* Data Bases (WDB) and on the corresponding querying to WDB. Let us stress that it is not only about databases with an access via Web. The database itself should be organized in the same way as Web. I.e. it must consist of hyperlinked pages distributed among the computers participating either in global network like Int*e*rnet or in some local, isolated from the outside world specific network based essentially on the same principles, except globality, and called also Intr*a*net [15].

This approach is based on a work on Bounded Set Theory (BST) and on its Δ-language [27]–[33], [19, 20] considered as a query language for data bases with complex or nested data. The notion of PTIME and LOGSPACE computability of set-theoretic operations was defined there in terms of graph transformers. It was used representation of hereditarily-finite *well-founded* sets (of sets of sets, etc.) by vertices of finite graphs (possibly labelled [30] and acyclic, except [31, 20]). Another source is P. Aczel's *non-well-founded* set theory [2, 4, 9, 24, 31] which allows considering "cyclic" sets and data represented by cyclic graphs.

Here we reintroduce in a natural way non-well-founded version of BST and Δ [31, 33] starting from motivations in WWW or WDB, rather than in abstract sets. We consider WDB as a (finitely branching) graph with labelled edges in a way analogous to [1, 22, 25], however, with an essential difference based on the above mentioned approach to consider graph vertices as (denoting) *antifounded sets*, called also *hypersets* [4][1]. To this end, the well-known notion of graph bisimulation due to D. Park (cf. [23]) is used crucially. In this respect our approach is closer to that of [18, 5] also based on bisimulation of graphs representing possibly cyclic "unstructured" databases. (Cf. more comments in Sect. 5)

On the other hand, this paper aims to give a natural interpretation of abstract hypersets and Δ, as a corresponding query language, in more realistic terms of WWW and WDB and to demonstrate both to theoreticians and to more practically oriented researchers possible applied value of hypersets. (Cf. [4] for other applications of hypersets to Computer Science and Philosophy.) This also dictates some "intermediate" form of presentation of the material. Properly speaking, it is given not a final set-theoretic description of the Web as it is, but rather some concise approach relating the corresponding ideas.

Mathematical results (Sect. 6) consist (i) in defining in terms of Δ (and also FO + IFP) a linear ordering on hereditarily-finite hypersets (respectively, on finite strongly extensional graphs) and (ii) in using this fact to characterize Δ-definable operations over hypersets exactly as PTIME-computable ones. The proofs for a pure (label-free) version of Δ are given in [20] on the base of [31].

[1] We will see that the abstract notion of hyperset occasionally and happily correlates also with that of *Hyper-Text*: "Web documents are created by authors using a language called HTML (HyperText Markup Language) that offers short codes (also called tags) to designate graphical elements and links. Clicking on links brings documents located on a server to a browser, irrespective of the server's geographic location." (From the Netscape Handbook.)

2 Web-like Data Bases

We define a (*state* or *instance* of) World-Wide Web or a *Web-like Data Base* (WDB) as a labelled graph (abbreviated $\mathcal{LG}$; it is also called *labelled transition system* [4]). See a motivation below. Formally, $\mathcal{LG}$ is a map $g : |g| \to \mathcal{P}(L \times |g|)$ with $|g|$ a set of *vertices*, $L = L^g$ a set of *labels* or *attributes* (from some given large class of labels $\mathcal{L}$) and $\mathcal{P}(|g|)$ the powerset. We consider vertices and labels quite abstractly, but *represent* or *encode* them as finite strings in a fixed finite alphabet. All the computations over a WDB will be *relativized* to its graph g considered as an "oracle" set-valued function. Three important subclasses of $\mathcal{LG}$ are *finite*, *finitely-branching* and *countably branching* labelled graphs ($\mathcal{FLG}$, $\mathcal{FBLG}$ and $\mathcal{CBLG}$). We write $l : u \in^g v$ or $g \models l : u \in v$ instead of $\langle l, u \rangle \in gv$ and say this as "u is an l-element of v in the sense of g".

Considering vertices u, v of a graph g and the relation $\in^g$, respectively, as *formal names of sets* (consisting of labelled elements) and as the *membership relation* between these sets might seem rather artificial for arbitrary graphs, however we will give in Sect. 3 more arguments for such way of thinking.

Motivations for "WDB as Graph".

Which way such a graph g is related to the Web? Let us think that vertices $u, v, \ldots$ are formal expressions like `http://www.botik.ru/PSI/AIReC/logic/`, i.e. addresses (Uniform Resource Locators, URLs, in a computer network) of WWW-pages. Then a *page*, which an address v refers to, is just the corresponding set of *references* $gv = \{l_1 : u_1, l_2 : u_2, \ldots\} \subseteq L \times |g|$. Formally "$l_i : u_i$" is an ordered pair, a vertex u_i labelled by l_i. In a real *finitely-branching* WDB this page may be represented as a finite file (or as a part of a file containing many such pages) which consists of the lines "$l_i : u_i$" listed in *any* order. This file is saved in one of the computers participating in the distributed WDB network. (Of course, there might be just one computer for the whole WDB.) The computer and file may be located by the address v. In the *infinitely-branching* case an "infinite page" would be generated line-by-line by a program. All these details are implicit in the program, a *browser*, which computes g on the base of the current state of all the computers in the Internet (or Intranet) participating eventually in the computation of g.

As in the real Web, it is reasonable if any page looks like a list of lines "l_i" consisting of labels only so that the corresponding hidden addresses "u_i", whose precise shape is not so relevant to a human user, are *invisible* on the screen display. *This idea of visibility is crucial for the future considerations.* We consider that labels play the role of atomic or primitive data or attributes, as in a "flat" relational DB, whereas addresses only organize these data into a complex/nested/graphical structure, possibly with cycles. By "clicking" mouse button on some occurrence of a label "l_i", we invoke the function g applied to the corresponding hidden address "u_i" to get visible a new page gu_i, etc. This process of "browsing" through the Web also suggests the "inverse" direction of (labelled) edges in the graph g defined as $u \stackrel{l}{\leftarrow} v$ iff $l : u \in^g v$.

According to this point of view on the nature of a Web page $p = \{l_1 : u_1, l_2 : u_2, \ldots\}$, its "visible part" is considered as *unordered multiset of words* $[\![p]\!] \rightleftharpoons \{\!\{l_1, l_2, \ldots\}\!\}$, i.e. just of labels l_i, *rather than a text*, i.e. ordered sequence of words. If a label l_i, as a word (or a picture), contains "enough" information then it is "non-clickable" ($g(u_i) = \emptyset$). Of course, any text may be represented as a very long word. E.g. one label on a page may be a longer text in which other shorter labels are mentioned in some way so that you can decide *informally* for yourself which of these labels must be clicked next. So, this representing data only via labels (which are appropriately organized via addresses) is sufficiently adequate and flexible abstract approach to Web-like Data Bases.[2]

If all the labels l_i in $gv = \{l_1 : u_1, l_2 : u_2, \ldots\}$ coincide then v actually defines a "uniform" set of elements $\{u_1, u_2, \ldots\}$. If all l_i are different then the page gv is a *tuple* or *record* with the *attributes* or *fields* l_i. (So, in a sense we need no special type of records.) Otherwise it could be called *quasituple* or *quasirecord* with u_i being l_ith *projection* of this quasituple. Let in general $v.l \rightleftharpoons \{u_i | l_i : u_i \in gv \,\&\, l_i = l\} = \{u_{i_1}, u_{i_2}, \ldots\}$. In the case of tuples this gives a *singleton set* $v.l_i = \{u_i\}$ (what is slightly different from the tradition to define $v.l_i = u_i$). If all gu_i are (quasi)tuples then gv is considered as *(quasi)relation* (a possibly heterogeneous set of (quasi)tuples). We see that the ordinary *relational* approach to Databases is easily absorbed by this WDB approach, and the same for the case of *nested relations* and *complex objects* (cf. also [30]) even having *cyclic structure*.

The World-Wide Web seems so widely known that there is no strong reason to present any illustrating examples. Rather, WWW is serving as a nice illustration and possible application of Hyper Set Theory whose "bounded" version we describe below. Say, there is nothing unusual in the possibility of cycles in the graph g representing a current instance of WWW (WDB).

3 Denotational Semantics of WDB via Hypersets

Now, the key point of the present approach to WWW or WDB is in defining *denotational semantics* $[\![v]\!]^g$ for any *vertex* v of a graph g in terms of visible part of a *page* $[\![p]\!]$ considered above. (Such a semantics $[\![\text{-}]\!]$ was also called *decoration* in [2] for the case of arbitrary graphs g; cf. also [4, 9, 31]. For well-founded g it is just A. Mostowski's *collapsing* [3] which we used in [27]–[30], [33]; cf. also [6]). Different vertices u and v may have the *same meaning*:

$$u \approx^g v \rightleftharpoons [\![u]\!]^g = [\![v]\!]^g. \qquad (1)$$

For example, it may be the case that $gu = gv$. (Just create or copy on your server a Web page which already exists somewhere else in the Web and assign it the corresponding new address.) These two addresses may be reasonably considered as equivalent by referring to exactly the same page. More generally, the pages (as sets of pairs) gu and gv may be

[2] "Isn't this like reasoning about a city by its tramway plan?"—asked one reader skeptically. Yes, almost such. Extend this plan by pedestrian schema, plans of some buildings, etc. The real Web demonstrates that this is no problem at all.

different, but *visibly coinciding* (as multisets of labels): $[\![gu]\!] = [\![gv]\!]$. Actually, $u \approx^g v$ means that it is impossible to find any difference between the addresses u and v if to *take into account only visible part* of corresponding arising pages during the multiple repeated process of browsing started respectively from u and v. However this seems intuitively rather clear, a formal definition is necessary; cf. (4). This approach of *neglecting the precise form of addresses* of the Web pages is quite natural because they play only the role of references or *hyperlinks*—no less, but also no more.

Alternatively and equivalently, membership-like notation ϵ^g suggests to consider $[\![v]\!]^g$ as a *set of sets of sets, etc.* defined recursively by the identity

$$[\![v]\!]^g = \{l : [\![u]\!]^g | l : u \,\epsilon^g\, v\}. \tag{2}$$

For example, if g contains exactly five (non-labelled) edges $u \,\epsilon^g\, v$, $u \,\epsilon^g\, v'$, $u \,\epsilon^g\, w$, $v \,\epsilon^g\, w$ and $v' \,\epsilon^g\, w$ then $[\![u]\!]^g = \emptyset$, $[\![v]\!]^g = [\![v']\!]^g = \{\emptyset\}$, and $[\![w]\!]^g = \{\{\emptyset\}, \emptyset\}$. If g is any *well-founded* $\mathcal{CBLG}$, i.e. has no infinite chains of vertices satisfying $\ldots \,\epsilon^g\, v_{i+1} \,\epsilon^g\, v_i \,\epsilon^g\, \ldots \,\epsilon^g\, v_1 \,\epsilon^g\, v_0$ or, equivalently, has no (directed) *cycles*, for the case of finite g, then $[\![v]\!]^g$ is "calculated" similarly for any $v \in |g|$ as a hereditarily-countable well-founded set. However, in non-well-founded case, such as in the graph , there is a problem. We need sets like $\Omega = \{\Omega\}$. This leads us to rather unusual non-well-founded or even *antifounded* or *hyper* set theory where some sets may serve as their own members, etc. [2, 4].

We confine ourselves to a subuniverse $\mathrm{HCA}_{\mathcal{L}}$ (for some supply of labels $\mathcal{L}$) of this set theory containing only *hereditarily-countable* (or finite) *hypersets*. Besides countability of each set $x \in \mathrm{HCA}$, it must satisfy the following *Anti-Foundation Axiom* (AFA) which is just a *finality property* in the category $\mathcal{CBLG}$:

> *For any* $\mathcal{CBLG}$ g *there exists a unique denotational semantics map* $[\![\text{-}]\!]^g : |g| \to \mathrm{HCA}$ *satisfying the identity (2) with respect to the membership relation* $\in$ *understood in the sense of* HCA *(i.e. as* $\in^{\mathrm{HCA}}$*).*

Evidently, HCA (if it exists) is determined by this axiom uniquely, up to isomorphism. According to the tradition of non-well-founded set theory, we will use $\in$ with no superscript HCA because we "believe" (or just behave as believing) in "real existence" of such a universe in the same way as for the "ordinary" well-founded universe, say, HC or HF of hereditarily-countable, respectively, -finite) sets. If g ranges only over finite, respectively, finitely branching graphs then $[\![v]\!]^g$ for $v \in |g|$ will range over the corresponding subuniverses $\mathrm{HFA} \subset \mathrm{HFA}^\infty$ of HCA.[3] While HFA consists of hereditarily-finite sets x whose *transitive closure*

$$\mathsf{TC}(x) \rightleftharpoons \{l : z | \exists n \geq 0 \exists \bar{l}\bar{z}.(l : z \in z_1, l_1 : z_1 \in \ldots \in z_n, l_n : z_n \in x)\} \tag{3}$$

[3] There are used also alternative denotations HF_0 for HF, $\mathrm{HF}_{1/2}$ for HFA and HF_1 for HFA^∞ [4] (actually, in the case of *pure* sets which do not involve labels). Also, in [30] HFA denoted *well-founded* universe with 'A' staying for 'attributes' which are called labels here. In this paper 'A' means 'Anti-Foundation Axiom' or 'Aczel'.

is finite, HFA^{∞} consists also of hereditarily-finite sets, however, their transitive closure may be infinite (just countable). Also sets from the subuniverses $\mathrm{HF} \subseteq \mathrm{HFA}$ and $\mathrm{HC} \subseteq \mathrm{HCA}$ correspond to the vertices of finitely and, respectively, countably branching *well-founded* graphs. It can be proved in the line of [2]

Theorem 1 *There exists the unique, up to isomorphism, universe* HCA *satisfying Anti-Foundation Axiom, and analogously also for* HFA^{∞} *and* HFA.

We only show how to define alternatively the corresponding equivalence relation $\approx^g \subseteq |g|^2$ (1) in terms of the graph g only, without any mentioning the semantic map $[\![\text{-}]\!]$ and the universe HCA (or HFA, HFA^{∞}).

First note that according to the above recursive description (2) of $[\![\text{-}]\!]^g$ in the universe HCA the relation $\approx^g$ must evidently satisfy the equivalence

$$u \approx^g v \Leftrightarrow \& \begin{cases} \forall l : x \,\epsilon^g\, u \exists m : y \,\epsilon^g\, v(l = m \& x \approx^g y) \\ \forall m : y \,\epsilon^g\, v \exists l : x \,\epsilon^g\, u(l = m \& x \approx^g y). \end{cases} \tag{4}$$

Actually, (4) have been informally discussed above in terms of visible indistinguishability of u and v. Any binary relation $R \subseteq |g|^2$ satisfying (4) with $\approx^g$ replaced by R and $\Leftrightarrow$ replaced by $\Rightarrow$ is called a *bisimulation* on g. Evidently, bisimulations are closed under unions so that there exists a *largest* one on g. It can be proved also (essentially, by using only AFA, (1) and (2)) that the latter *coincides* with $\approx^g$. So, the reader may completely forget (if he prefers this) about the universes HFA, HFA^{∞} and HCA and work directly in terms of graphs and bisimulations. However, it would be rather unreasonable to ignore our everyday and fruitful set-theoretic experience also applicable to hypersets. On the other hand, graph representation of such hypersets is very useful because this allows to consider computability notions over these abstract universes of sets in terms of graph transformers.

Moreover, in finitely branching case $\approx^g$ may be obtained as intersection[4]

$$\approx^g = \bigcap_{i=0}^{\infty} \approx^g_i \quad \text{where} \tag{5}$$

$$\begin{aligned} u \approx^g_0 v \quad &\rightleftharpoons \quad \mathsf{true} \quad \text{and} \\ u \approx^g_{i+1} v \quad &\rightleftharpoons \quad \& \begin{cases} \forall l : x \,\epsilon^g\, u \exists m : y \,\epsilon^g\, v(l = m \& x \approx^g_i y) \\ \forall m : y \,\epsilon^g\, v \exists l : x \,\epsilon^g\, u(l = m \& x \approx^g_i y) \end{cases} \end{aligned} \tag{6}$$

so that $\approx^g_0 \supseteq \approx^g_1 \supseteq \ldots \supseteq \approx^g$. For the case of a finite graph g this sequence evidently stabilizes: $\approx^g_i = \approx^g$ for some i, so that $g \mapsto \approx^g$ is *computable* (in polytime wrt cardinality of $|g|$). For finitely branching graphs with infinite number of vertices we have only *semidecidability* of $\not\approx^g$ (wrt the "oracle" g). By considering graphs *up to bisimulation* (in contrast to, say, [1] devoted to infinite finitely branching graphs as WWW instances), this is the main technical reason why we need some additional efforts for developing corresponding approach for this infinite case (i.e. for the case of HFA^{∞}; corresponding paper is in progress.) So, we concentrate here mainly on the finite case (HFA).

[4] In general case i must range over *ordinals*, instead of natural numbers.

Now, it is natural to consider the "membership" relation ϵ^g *up to bisimulation* $\approx^g$ so that $\approx^g$ proves to be a *congruence* relation on $|g|$:

$$l : u \in^g v \rightleftharpoons \exists l' : u' \epsilon^g v(l = l' \& u \approx^g u'), \qquad (7)$$
$$u \approx^g u' \& v \approx^g v' \Rightarrow (l : u \in^g v \Leftrightarrow l : u' \in^g v').$$

Also the ordinary *extensionality axiom* written in the form (like (4))

$$u = v \Leftrightarrow \& \begin{cases} \forall l : x \in u \exists m : y \in v(l = m \& x = y) \\ \forall m : y \in v \exists l : x \in u(l = m \& x = y) \end{cases} \qquad (8)$$

holds in any g if symbols $=$ and $\in$ are interpreted as $\approx^g$ and, respectively, as $\in^g$ (i.e. as ϵ^g considered up to $\approx^g$). Actually, the following *axiom of strong extensionality* holds in the quotient $g/_{\approx^g}$ or in HCA, HFA$^\infty$, HFA:

$$xRy \Rightarrow x = y \quad \text{for arbitrary bisimulation relation R wrt } \in.$$

Graph g is called *strongly extensional* ($\mathcal{SE}$) or *canonical* if $\approx^g \, = \, =^g$. It follows also that the vertices of any (countable, $\mathcal{SE}$) WDB g and the relation $\in^g$ corresponding to the edges of g (up to $\approx^g$) may be identified with corresponding HCA-sets and, respectively, with the membership relation $\in^{\text{HCA}}$ for these sets.

We conclude that the Web or any WDB (considered statically) may be naturally interpreted as an "initial" (*transitive*) part of the universe HCA or HFA$^\infty$ or HFA, depending on its type: countable, finitely branching, or just finite WDB.

WDB Schemas.

Different instances g, g' of WDB may have the same *schema* or a *type* s which is also presented as a (desirably $\mathcal{SE}$) graph $s : |s| \to \mathcal{P}(\mathcal{L} \times |s|)$. Given arbitrary binary relation on labels $T \subseteq \mathcal{L}^2$, consider corresponding largest, *relativized to* T*, bisimulation relation* (now denoted as $::_T^{g,s}$; cf. [30])

$$u ::_T^{g,s} v \Leftrightarrow \& \begin{cases} \forall l : x \,\epsilon^g\, u \exists m : y \,\epsilon^s\, v(lTm \& x ::_T^{g,s} y) \\ \forall m : y \,\epsilon^s\, v \exists l : x \,\epsilon^g\, u(lTm \& x ::_T^{g,s} y), \end{cases}$$
$$g ::_T s \rightleftharpoons \forall u \in |g| \exists v \in |s| (u ::_T v) \& \forall v \in |s| \exists u \in |g| (u ::_T v).$$

(Alternatively, we could consider, as in [30], that only the first conjunct is present in these formulas. In this case we should rather call $::$ *simulation* relation.)

Evidently, $\approx \, = \, ::_{=_\mathcal{L}}$, $::_{T_1 \circ T_2} = (::_{T_1}) \circ (::_{T_2})$ and $::_{T^{-1}} = (::_T)^{-1}$. If $T \subseteq \mathcal{L}^2$ is transitive (respectively, reflexive, symmetric) then such is $::_T$.

Let now lTm denote some typing relation "l *is of type* m" between labels like: "`Pereslavl-Zalessky` *is of type* `city`", "`Moscow` *is of type* `capital`", "`Moscow` *is of type* `word`", "`word` *is of type* `word`", "`5` *is of type* `integer`", etc. Then $g ::_T s$ also means that g *is of type* s or that s is a *schema* of the database (state) g. If $T \subseteq \mathcal{L} \times \mathcal{L}$ is a (partial) map $\mathcal{L} \to \mathcal{L}$ then such s is defined uniquely, up to $\approx$, and may be easily obtained from g by replacing in g all labels by their T-images.

Any given typing relation $T \subseteq \mathcal{L}^2$ also induces corresponding typing relation $x ::_T \alpha$ between arbitrary HFA-sets x and α. Any $\alpha \in$ HFA may serve as a type. However, for T a map, a type α is usually simpler than x, and it properly reflects the "nested" structure of x. Cf. also [30] for corresponding tabular representations of (well-founded) HF-sets according to their types. This makes the present approach related with (nested) relational databases. Note, that now both sets and their types may be circular.

Isomorphism of WDBs up to Bisimulation.

Let T, now denoted as $\simeq$, be a bijection $L \to L'$ between labels of two WDB graphs g and g'. Then corresponding relation between vertices $::_T^{g,g'}$, denoted as $\simeq^{g,g'}$, proves to be an *isomorphism, up to bisimulation*, between "initial parts" of g and g' because

$$
\begin{array}{l}
u \simeq^{g,g'} u' \,\&\, v \simeq^{g,g'} v' \,\&\, l \simeq l' \Rightarrow \\
\qquad (l : u \in^g v \Leftrightarrow l' : u' \in^{g'} v')\&(u \approx^g v \Leftrightarrow u' \approx^{g'} v')
\end{array}
$$

holds for all $u, v \in |g|$, $l \in L$ and $u', v' \in |g'|$, $l' \in L'$. We usually write $u \simeq u'$ instead of $u \simeq^{g,g'} u'$ to simplify notation and consider this as *three* place relation between u, u' *and* a bijection $\simeq \subseteq L \times L'$. If there is some predefined relational structure on labels such as a linear order $\prec_{\mathcal{L}}$, a typing relation, etc. (considered as primitive predicates of the Δ-language below) then we assume additionally that $\simeq$ is the ordinary *first-order isomorphism* between L and $L' \subseteq \mathcal{L}$.

4 Δ-Language for Hypersets

It is widely recognized high expressive power and flexibility of the classical set theory. Let us present the syntax of Δ-*language* for some its *bounded* version:

$$
\begin{array}{rcl}
\Delta\text{-formulas} & ::= & l = m \mid P(\bar{t}) \mid \varphi \& \psi \mid \varphi \vee \psi \mid \neg\varphi \mid \forall l : x \in s.\varphi \mid \exists l : x \in s.\varphi \mid \\
& & [\textsf{the-least}\; P.(P(\bar{x}) \Leftrightarrow \theta(\bar{x}, P))](\bar{t}) \\
\Delta\text{-terms} & ::= & \text{set-variables } x, y, \ldots \mid \emptyset \mid \{l_1 : t_1, \ldots, l_k : t_k\} \mid \\
& & \bigcup\{t(l, x) | l : x \in s \& \varphi(l, x)\} \mid \textsf{TC}(s) \mid \textsf{D}(s, t)
\end{array}
$$

Here $t, \bar{t}, s$ are Δ-terms (or lists of terms); φ, ψ are Δ-formulas; l, m, l_i are label variables which formally are not Δ-terms; P is a predicate variable (only of set arguments); variables x and l are not free in s; θ is Δ-formula which involves (at any depth) only atomic terms; and all occurrences of the predicate variable P in θ are *positive*. Also there may be some additional primitive predicates over labels, besides the *equality* $l =_{\mathcal{L}} m$, such as some given *linear order* $l \prec_{\mathcal{L}} m$, or the predicate "*l is a substring of m*" or discussed above some typing relation.

The meaning of the most of these constructs in the universe HFA (or even in HCA) is straightforward. $\bigcup\{t(l,x)|l : x \in s\&\varphi(x,l)\}$ is the *union* of the family of sets $t(l,x)$ where x and l satisfy the condition after "|". The *transitive closure* TC has been defined in (3). [$\mathsf{the\text{-}least}$ $P.(P(\bar{x}) \Leftrightarrow \ldots)$] denotes the least predicate on the universe HFA such that the corresponding equivalence holds for all $\bar{x} \in$ HFA. It evidently exists by the positivity requirement on P and may be obtained as *union* $\bigcup_i P_i$ with $P_0 \rightleftharpoons \mathsf{false}$ and $P_{i+1}(\bar{x}) \rightleftharpoons \theta(\bar{x}, P_i)$). Note, that this union is actually "locally finite" due to $\theta(\bar{x},\bar{y},P) \Leftrightarrow \theta(\bar{x},\bar{y},P\restriction\mathsf{TC}(\{\bar{x},\bar{y}\}))$ where $\bar{x},\bar{y}$ are all free set variables of θ (which has no complex terms!). This construct is actually equivalent in HFA (essentially as in [14]) to its *inflationary* version [$\mathsf{inflationary}$ $P.(P(\bar{x}) \Leftrightarrow (P(\bar{x}) \vee \theta(\bar{x},P))](\bar{t})$ with no requirement of positivity on P in θ.

Unordered sets may encode *ordered triples* of the kind

$$\langle l,u,v\rangle \rightleftharpoons \{l : \{l : u\}, l : \{l : u, l : v\}\}. \tag{9}$$

Then arbitrary set $s \in$ HFA serves also as a finite graph by taking all ordered triples $\langle l,u,v\rangle \in s$ (if such exist) as the labelled edges of this graph. Therefore for *decoration operation* D we may postulate the axiom analogous to (2) for $[\![\text{-}]\!]$:

$$\mathsf{D}(s,v) = \{l : \mathsf{D}(s,u)|l : \langle l,u,v\rangle \in s\}.$$

Decoration is the only construct of this language which has not a meaning in general for the ordinary well-founded universe HF. However, this is a natural operation which, given arbitrary (finite) graphical, possibly cyclic "plan" $s \in$ HFA, constructs a set $\mathsf{D}(s,v) \in$ HFA according to this plan.

Note that strong extensionality allows to define in Δ equality relation $u = v$ between HFA-sets (missing in the present version of Δ) recursively, as the largest fixed point via (4) or (8) (easily reducible to $\mathsf{the\text{-}least}$ construct) by using primitive equality $l =_{\mathcal{L}} m$ on labels. Therefore $\in^{\mathrm{HFA}}$ is also Δ-definable by using bounded quantification as $l : u \in v \rightleftharpoons \exists m : w \in v(l = m\&u = w)$.

The origins of Δ-language go back to the well-known notion of Δ_0-formulas of A. Lévy and to so called *basic* = *rudimentary* set-theoretic operations of R.O. Gandy [10] and R.B. Jensen [17]. Note also somewhat related theoretical approach to SETL in [6, 7] and set-theoretic programming language STARSET [11]. The latter, in comparison with SETL and Δ, does not use deeply nested sets. So called semantic or Σ-programming [12] is based mainly on considering unbounded, unlike Δ, positive existential quantification.

Some label-free version of this Δ-language for pure universe HFA was considered in [27], as well as labelled one for the well-founded case in [30]. Natural *axioms* describing the meaning of Δ-definable operations and predicates are presented in [27, 28, 33] and [31, 33] for pure well-founded and, respectively, non-well-founded case, as well as corresponding proof-theoretic considerations, so that we actually have a *Bounded Set Theory with Anti-Foundation Axiom*, BSTA. It may be used, say, for proving correctness/equivalence of Δ-queries.

5 Querying and Computing over WDB

Any Δ-term $t(\bar{l},\bar{x})$ or Δ-formula $\varphi(\bar{l},\bar{x})$ defines a map, called also *query*, $\mathrm{HFA}^k \to \mathrm{HFA}$ or, respectively, $\mathrm{HFA}^k \to \{\mathsf{true},\mathsf{false}\}$. (We represent any label argument $l_i \in \mathcal{L}$ by the set $\{l_i : \emptyset\} \in \mathrm{HFA}_{\mathcal{L}}$.) Such a map $q : \mathrm{HFA} \to \mathrm{HFA}$ is called (PTIME) *computable* by a graph transformer $Q : \langle g, v\rangle \mapsto \langle g', v'\rangle = Q(g,v)$ if

$$q([\![\langle g,v\rangle]\!]) = [\![Q(g,v)]\!] \quad (\text{where} \quad [\![\langle g,v\rangle]\!] \rightleftharpoons [\![v]\!]^g) \tag{10}$$

holds for all finite g and $v \in g$, and Q is (PTIME) computable. We may also assume that g' extends the graph g by some "temporary" vertices and edges used to "calculate" the result $[\![\langle g',v'\rangle]\!] = q([\![\langle g,v\rangle]\!])$ without changing the meaning of the old vertices in g. We also abbreviate $\langle g',v'\rangle = Q(g,v)$ as $v' = Q(v)$.

It is natural to call q and Q *abstract* and *concrete* versions, respectively, of a query to WDB g, or as its *denotational* and *operational semantics*. The specific way of calculating a query Q which guarantees the identity (10) consists in

- multiple repeated *browsing*, i.e. applying the function g, starting from an input address v of a page,
- *searching* the labels on a current page satisfying some (also computable) condition (e.g. on containing by a text-label some substring(s) of symbols),
- *(re)organizing* this information according to the query Q by *creating new pages* with the labels found above and *assigning them some addresses*,
- taking into account during the computation *only visible part* of pages,
- *outputting* as the final result (a *page* with) some *address* v' which, together with all (possibly new) pages accessible from it via hyperlinks will constitute the whole answer to the query.

In particular, this is the case for arbitrary Δ-definable queries/updates $Q(\bar{v})$ to WDB g where now set variables $\bar{v}$ range over addresses (i.e. graph vertices) rather than over corresponding HFA-sets (cf. also [31]). Here are examples.

1. Given labels $l_1,\ldots,l_n$ and addresses $v_1,\ldots,v_n$ ($n \geq 0$), create a new page $\{l_1 : v_1,\ldots,l_n : v_n\}$ (i.e. a new file on a computer in the network with the corresponding address) consisting of the lines $l_1 : v_1,\ldots,l_n : v_n$.
2. Take the *union* $v_1 \cup v_2 = \bigcup\{v_1,v_2\}$ of two pages given by their addresses v_1 and v_2 by concatenating corresponding files. The same for more general union construct $\bigcup\{t(l,x)|l : x \in s\}$, assuming that we can compute $t(l,x)$.[5]

[5] Let us postulate additionally that during all the subcomputations $t(l,x)$ for $l : x \in gs$ the sets of newly created addresses (in all the intermediate steps) for different pairs $l : x$ do not intersect. This allows to accomplish all these subcomputations independently/concurrently, with no conflicts between new addresses, and actually guarantees polytime complexity of the resulting query. Cf. Theorem 3 (a) below.

3. Bounded quantifiers $\forall l : x \in s.\varphi$ and $\exists l : x \in s.\varphi$ and the general union $\bigcup\{t(l,x)|l : x \in s\&\varphi(l,x)\}$ are also based on looking through pages having references $l : x$ in gs and satisfying given (computable) condition $\varphi(l,x)$.

4. To compute the-least construct use its local property (cf. Sect. 4).

5. To compute decoration $\mathsf{D}(s,w)$ by any two given page addresses s and w

   (a) start depth 3 browsing from s to gather (actually taking into account bisimulation which is itself computable by the-least construct) all the "triples" $l : \langle l,u,v\rangle$ (9) from the page gs,

   (b) consider each this triple as a l-labelled edge of a graph γ,

   (c) create new (i.e. not used in the WDB instance g) different addresses NewAddr(u), NewAddr(v), NewAddr(w), ... for all the vertices of γ,

   (d) extend g by creating new pages with these new addresses to imitate γ,

   (e) output NewAddr(w) as the final result.

   (We omit some more details.)

We see that Δ is not only a *pure* set-theoretic language over HFA whose meaning may be understood quite abstractly, but simultaneously a "graph" language interpreted solely in terms of the computation over the Web or WDB.

Unlike HFA, when working in terms of a WDB state g the ordinary equality and g-membership relations between addresses $u = v$ and $l : u \epsilon^g v$ *cannot* be considered here as legal Δ-queries. Otherwise, we would be able to distinguish visibly non-distinguishable addresses. However, bisimulation $u \approx^g v$ (4) and another membership relation of the kind $l : u \in^g v$ (7) have been actually Δ-defined. Fortunately, "membership" relation of the form $l : u \epsilon^g v$ *is allowed in the context of bounded quantification* which is effectively computable by looking through the finite page gv.

Database Integrity Constraints

also may be expressed in Δ-language. A trivial example: for $x \in$ HFA to be a tuple (vs. quasituple, i.e. an arbitrary HFA-set) is formulated as the constraint $\forall l : y, l' : y' \in x(l = l' \Rightarrow y = y')$. We may describe that some relations (sets of tuples) are of a specified arity, or postulate some functional dependencies or some interleaving between tuples and relations when going deeper and deeper into a (possibly cyclic) set, etc.

Important Properties of Δ-Definable Queries.

It can be shown by induction on the definition of Δ-language that (in terms of its operational semantics in WDB terms sketched above) all Δ-definable queries *preserve bisimulation relation $u \approx^g v$ between (addresses of) pages* (cf. [31] for more details). This means that any page query t and boolean query φ should not distinguish between pages with the same meaning:

$$u \approx^g v \Rightarrow t(u) \approx^g t(v) \& (\varphi(u) \Leftrightarrow \varphi(v)).$$

Evidently, all Δ-queries *preserve (do not create new) labels* in the sense that any label participating in $\mathsf{TC}(t(\bar{l},\bar{u}))$ must participate also in $\{\bar{l}\} \cup \mathsf{TC}(\{\bar{u}\})$.

A very important property of Δ-definable queries consists in their *genericity* (or *abstractness*), **up to bisimulation**. We define this notion following [30] where it was applied directly to HF-sets rather than to representing (finite acyclic) graphs. Cf. also [1] where the notion of genericity is considered only for boolean queries and without any connection to bisimulation, unlike this paper and [30].

A query $Q(u)$ is called *generic (up to bisimulation)* if it preserves (i) bisimulation, (ii) labels and also (iii) isomorphism, up to bisimulation, of its inputs. I.e. for any two graphs g and g' and an isomorphism $\simeq$ between corresponding sets of labels L and L' (and therefore between some vertices in $|g|$ and $|g'|$)

$$u \in |g|,\ u' \in |g'|,\ u \simeq u' \quad \text{implies} \quad Q(u) \simeq Q(u')$$

(according to the *same* bijection $\simeq$ between labels!).

These properties of queries appear as quite natural, even independing on the above Δ-language of queries. Moreover, considering properly the *computability notion* of a query as a graph transformer $Q : \langle g, v\rangle \mapsto \langle g', v'\rangle = Q(g, v)$ satisfying (10) (reasonably defined by using encoding of *abstract* labelled graphs; cf. also [30]) leads *inevitably* to genericity of Q.

Some Comparisons.

Of course, it is not strictly necessary to support set-theoretic (based on bisimulation) level of abstraction if paying attention to the precise syntax of addresses; cf. [1, 22, 25]. E.g. in [1] a general Turing machine model of browsing and searching is discussed where genericity of the underlying finitely-branching graph is understood in terms of the ordinary isomorphism of graphs, what corresponds to understanding addresses of the Web pages essentially more literally, than here. Bisimulation and corresponding approach to querying for the graphs representing data has been discussed in

- [27]–[30],[33]: the acyclic case, via decoration restricted to collapsing, including a more general approach to encoding of HF-sets and to PTIME-computability over HF; cf. especially labelled case [30],
- [32, 19]: approaching to LOGSPACE-computability over HF,
- [31, 33]: cyclic label-free case on a more abstract theoretical level than here, including some proof theoretic considerations on BST, AFA, and a Logic of Inductive Definitions,
- and also in the papers of A. Kosky [18] and P. Buneman, et al [5].

The main informal difference of our approach with [18, 5] is the set-theoretical point of view based on the observation that the graphs and operations over them considered up to bisimulation are essentially (non-well-founded) sets of sets of sets, etc., and well-known set-theoretical operations, respectively. Also note, that query languages UnQL and UnCAL discussed in [5] and formulated in terms of graphs may be essentially (up to some unclear places in syntax and semantics) embedded in Δ (if it is considered as a language of graph transformers). However, the decoration operation D of Δ is not definable there and also in SRI [18]. Any operation of these languages preserves the well-foundedness of input data, i.e. being applied to the acyclic data structure results also in the acyclic one, while D obviously does not. (An argument to D may be cyclic graph—something like a syntactic specification of a data—*represented* by an HF-set, but all HF-sets themselves *are* of "acyclic" nature.) It follows that these languages do not define all PTIME computable queries, unlike Δ (cf. Theorem 3 below).

6 Linear Ordering of HFA and PTIME Computability

Let, for a while, a finitely branching graph g (say, = HFA or HFA$^\infty$) contain no labels. Then we can define iteratively as in [20] (by appropriate reconstructing a formula from [30] with using an idea from [8]) strict linear preorders $\prec_k$ on g (which are decidable uniformly on k with respect to the oracle g):

$$
\begin{aligned}
& x \prec_0 y \rightleftharpoons \mathsf{false}, \\
& x \prec_{k+1} y \rightleftharpoons x \prec_k y \vee \\
& \qquad \{x \approx_k y \mathbin{\&} \exists u \in y[\forall v \in x(u \not\approx_k v) \mathbin{\&} \\
& \qquad \forall w \in x \cup y(u \prec_k w \to \exists p \in x \exists q \in y(p \approx_k w \mathbin{\&} q \approx_k w)]\}.
\end{aligned}
$$

This definition may be easily adapted for the case of labelled graphs with some predefined strict linear order $\prec^L$ on labels. Note, that expressions like $x \approx_k y$ may be replaced by actually equivalent formulas $x \not\prec_k y \& y \not\prec_k x$. It follows that

Theorem 2 *The "limit" strict linear order* $\prec \rightleftharpoons \bigcup_i \prec_i$ *on* HFA *(and even on* HFA$^\infty$*) is definable in* Δ *relative to* $\prec_{\mathcal{L}}$ (both by the inflationary and positive version of $\mathsf{the\text{-}least}$ construct of Δ). *The same holds for vertices of any finite or finitely-branching* $\mathcal{SE}$ *graph with linear ordered labels and FO+LFP (or IFP).*

Remember that FO+LFP (or IFP) is First-Order Logic + the corresponding logical analogue of $\mathsf{the\text{-}least}$ (respectively, $\mathsf{inflationary}$) fixed point construct of Δ. Both versions are equivalent over finite structures [14]. Definability in Δ over HFA was reduced in [31] to definability of graph transformers in FO+LFP. It is well-known that such kind of recursion in *linearly ordered* finite domains, like a finite segment of natural numbers, is equivalent to PTIME-computability [26, 16, 21, 34, 13]. Therefore, Theorem 2 (cf. also [20]) implies

Theorem 3 *(a) Definability in FO + IFP (i.e. recursion) over finite strongly extensional labelled graphs with predefined linear order only on the labels is equivalent to PTIME computability over these graphs.*

(b) Assuming a linear order $\prec_{\mathcal{L}}$ on labels, Δ-definable operations over $\mathrm{HFA}_{\mathcal{L}}$ *= all (generic) PTIME-computable queries* HFA $\to$ HFA *(actually, relative to oracles corresponding to free predicate variables in Δ-query) under graph representations of sets according to the semantics* $[\![\text{-}]\!] : \mathcal{FLG} \to \mathrm{HFA}_{\mathcal{L}}$.

We conclude that set-theoretic and WDB query language Δ is *sufficiently complete and natural*. Also for the case of "small" finite WDB all the definable queries are, in a sense, "feasibly" computable. However, the latter note should not be overestimated. So, bisimulation/non-bisimulation is practically rather expensive procedure, however being PTIME computable. (It is PTIME-complete problem even for acyclic graphs [6].) On the other hand, it can be evaluated permanently, by some program, so that we could use partial results of its work, when needed. For the case of finitely branching graphs (and HFA^{∞}, unlike HFA) we can assert only that $\prec$ and $\not\approx$ (or $\neq$ in HFA^{∞}) are *semicomputable*.

Anyway, the present work is conceptual rather than practical one. We just attempted to present a uniform, concise set-theoretic view on querying to hyperlinked data.

References

[1] Abiteboul, S., Vianu, V.: Queries and computation on the Web. Database Theory—ICDT'97, 6th International Conference, Delphi, Greece, January 1997, Proceedings. Lecture Notes in Computer Science **1186** Springer (1995) 262–275

[2] Aczel, P.: Non-well-founded sets. CSLI Lecture Notes No. 14, 1988

[3] Barwise, J.: Admissible sets and structures. Springer, Berlin, 1975

[4] Barwise, J., Moss, L.: Vicious Circles: on the mathematics of circular phenomena, CSLI Lecture Notes, 1996

[5] Buneman, P., Davidson, S., Hillebrand, G., Suciu, D.: A query Language and Optimization Techniques for Unstructured Data. Proc. of SIGMOD, San Diego, 1996

[6] Dahlhaus, E.: Is SETL a suitable language for parallel programming?—a theoretical approach. Proc. of CSL'87, Lecture Notes in Computer Science **329** (1987) 56–63

[7] Dahlhaus, E., Makowsky, J.: The Choice of programming Primitives in SETL-like Languages. Proc. of ESOP'86, Lecture Notes in Computer Science **213** (1986) 160–172

[8] Dawar A., Lindell S., Weinstein, S.: Infinitary Logic and Inductive Definability over Finite Structures. Information and Computation, **119**, No. 2 (1995) 160-175

[9] Fernando, T.: A Primitive recursive set theory and AFA: on the logical complexity of the largest bisimulation. Report CS-R9213 ISSN 0169-118XCWI P.O.Box 4079, 1009 AB Amsterdam, Netherlands

[10] Gandy, R.O.: Set-theoretic functions for elementary syntax. Proc. Symp. in Pure Math. **Vol. 13, Part II** (1974) 103–126

[11] Gilula, M.M.: The Set Model For Database and Information Systems. Addison Wesley, 1994

[12] Goncharov, S.S., Ershov, Ju.L., Sviridenko, D.I.: Semantic programming, Kugler H.J., ed. Information Processing'86 , Elsevier, North Holland (1986) 1093–1100

[13] Gurevich, Y.: Algebras of feasible functions. Proc. 24th IEEE Conf. on Foundations of Computer Science (1983) 210–214

[14] Gurevich, Y., Shelah, S. : Fixed-point extensions of first-order logic. Annals of Pure and Applied Logic **32** (1986) 265-280

[15] Cf. `http://www.intranetjournal.com/`.

[16] Immerman, N.: Relational queries computable in polynomial time. Proc. 14th. ACM Symp. on Theory of Computing (1982) 147–152; cf. also Information and Control **68** (1986), 86–104

[17] Jensen, R.B.: The fine structure of the constructible hierarchy. Ann. Math. Logic **4** (1972) 229-308

[18] Kosky, A. : Observational properties of databases with object identity. Technical Report MS-CIS-95-20. Dept. of Computer and Information Science, University of Pennsylvania (1995)

[19] Lisitsa, A.P., Sazonov, V.Yu.: Δ-languages for sets and LOGSPACE-computable graph transformers, Theoretical Computer Science **175** (1997) 183–222

[20] Lisitsa, A.P., Sazonov, V.Yu.: On linear ordering of strongly extensional finitely branching graphs and non-well-founded sets. DIMACS TR 97-05. Rutgers University, February 1997 (`http://dimacs.rutgers.edu/TechnicalReports/1997.html`; a shorter version of this paper to appear in Proc. of LFCS'97, Springer LNCS)

[21] Livchak, A.B.: Languages of polynomial queries. In: Raschet i optimizacija teplotehnicheskih ob'ektov s pomosh'ju EVM, Sverdlovsk (1982) 41 (in Russian)

[22] Mendelzon, A. O., Mihaila, G.A., Milo, T. : Querying the World Wide Web. Draft, available by ftp: `milo@math.tau.ac.il` (1996)

[23] Milner, R. : Operational and algebraic semantics of concurrent processes. In: Handbook of Theoretical Computer Science, Ed. J. van Leeuwen, Elsevier Science Publishers, B.V. (1990) 1201–1242

[24] Mislove, M., Moss, L., Oles, F.: Non-Well-Founded Sets Modeled as Ideal Fixed Points. Information and Computation, **93** (1991) 16-54

[25] Quass, Y., Rajaraman, A., Sagiv, Y., Ullman, J., Widom, J.: Querying semistructured heterogeneous information. Technical Report, Stanford University, December 1995. Available by anonymous ftp from `db.stanford.edu`.

[26] Sazonov, V.Yu.: Polynomial computability and recursivity in finite domains. Elektronische Informationsverarbeitung und Kybernetik **16**, No. 7 (1980) 319–323

[27] Sazonov, V.Yu.: Bounded set theory and polynomial computability. Proc. of All Union Conf. on Applied Logic, Novosibirsk (1985) 188–191 (In Russian).

[28] Sazonov, V.Yu.: Bounded set theory, polynomial computability and Δ-programming. Application aspects of mathematical logic. Computing systems **122** (1987) 110–132 (In Russian). Cf. also a short English version of this paper in: Lect. Not. Comput. Sci. **278**, Springer (1987) 391–397

[29] Sazonov, V.Yu.: Bounded set theory and inductive definability. Abstracts of Logic Colloquium'90. JSL **56** Nu.3 (1991) 1141–1142

[30] Sazonov, V.Yu.: Hereditarily-finite sets, data bases and polynomial-time computability. Theoretical Computer Science, **119** (1993) 187–214.

[31] Sazonov, V.Yu.: A bounded set theory with anti-foundation axiom and inductive definability. Computer Science Logic, 8th Workshop, CSL'94 Kazimierz, Poland, September 1994, Selected Papers. Lecture Notes in Computer Science **933** Springer (1995) 527–541

[32] Sazonov, V.Yu., Lisitsa, A.P.: Δ-languages for sets and sub-PTIME graph transformers. Database Theory – ICDT'95, 5th International Conference, Prague, Czech Republic, January 1995, Proceedings. Lecture Notes in Computer Science **893** Springer (1995) 125–138

[33] Sazonov, V.Yu.: On Bounded Set Theory (10th International Congress on Logic, Methodology and Philosophy of Sciences, Florence, 1995), M.L.Dalla Chiara et al. (eds.), Logic and Scientific Methods, Kluwer Academic Publishers, 1997, 85–103

[34] Vardi, M.: Complexity of relational query languages. Proceedings of 14th Symposium on Theory of Computation (1982) 137–146