A sequence format and namespace for complex oligo- and polysaccharides

Version 3
„AOI“

15-05-07

S. Herget, R. Ranzinger, W.v.d.Lieth
Central Spectroscopic Department
DKFZ Heidelberg

Content

Table of contents

Update

Please take a look on *http://www.eurocarbdb.org/* for updates of this documentation.

Contact

If you have suggestions, questions or comments, do not hesitate to contact us:

W. von der Lieth & The Heretics
Central spectroscopic department
German Cancer Research Center
Heidelberg, Germany
w.vonderlieth@dkfz-heidelberg.de

Acknowledgments

We would like to thank all our numerous collaborators in the glycobioinformatics community for helpfull comments, fruitfull discussions and enlighting remarks. Special thanks belong to all members of the EuroCarbDB project, which showed endless readiness for valuable discussions. We would like to thank the following persons in special for critical reading of different versions of the manuscript and indispensable suggestions:

Dr. A. Ceroni	Dr. P. Toukach	Prof. Dr. J.P. Kamerling
Dr. K. Maaß	Dr. T. Luetteke	Dr. M. Frank

This work has been funded by the EU sixth framework programme and the German Research Foundation.

1. Introduction

This documentation describes a structured approach to capture the information typically contained in carbohydrate sequences. We define herewith three variants of a sequence format. One is a condensed, sorted code suited as an unique key for carbohydrate sequences in database applications (GlycoCT{condensed}). The more verbose XML – syntax contains the same information, but facilitates exchange of structural data and can be used for a description of bigger macromolecular assemblies in which carbohydrates are present (GlycoCT{XML}). The namespaces of all entities are controlled, with an emphasis on a **controlled vocabulary for carbohydrates**, which is fully machine-readable. A **canonical numbering** for both residues and linkages is an integral part of this format.

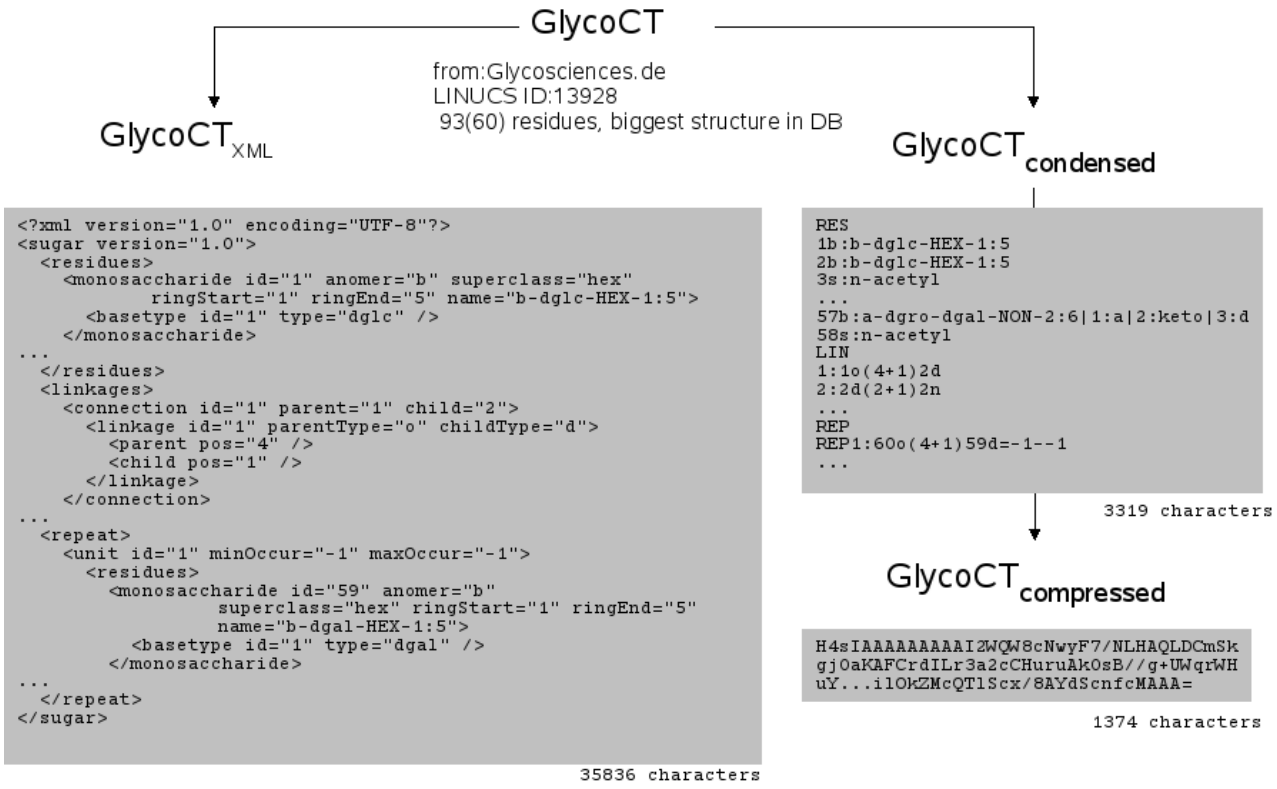


Figure 1: GlycoCT exists in three flavours. The XML variant facilitates parsing, information transfer and validation. The condensed form is easier to be read by humans and fully sorted. A compressed variant reduces the space needed by one third.

1.1 Theoretical considerations

Carbohydrate sequences can be probably best described as graphs with residues as vertexes and linkages as edges (Figure 2). As the sequences contain a preferred direction from the reducing end to the non-reducing end, the graphs can be viewed as directed (digraphs). The existence of potential multiple connections between two vertexes can degenerate the graph to a multigraph. The rare cyclisation of carbohydrate structures (e.g. dextrines) can lead to cyclic graphs. Repeating units can be modelled as special entities. Limited analytical techniques resulting in partial structure elucidation can produce uncertainties in the sequences, especially regarding the location of terminal nodes (capping units). Some secondary modifications (e.g. sulfatation) are present only on a fraction of the nodes of repeating units, leading to non-stoichiometric modification patterns. Alternative residues (labels) for a certain vertex can be the result of a structure elucidation.

Furthermore the sequences can contain incomplete linkage information or missing connections.

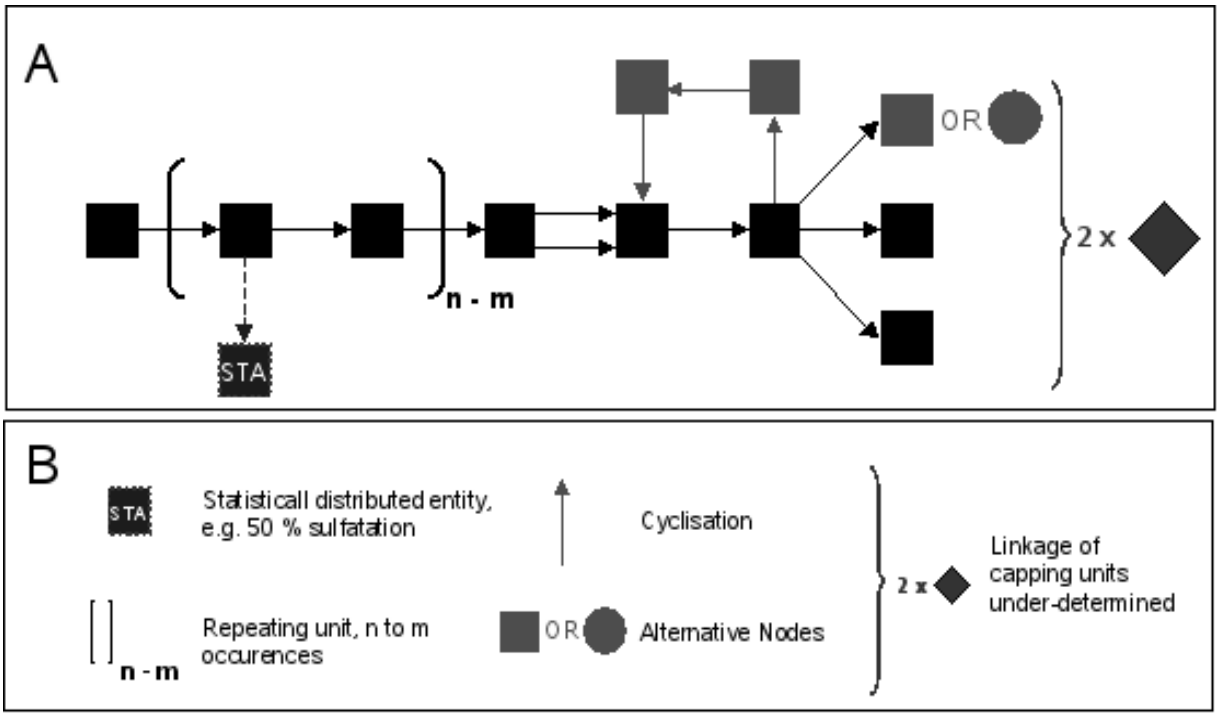


Figure 2: A generic schema of a complex carbohydrate sequence. (A) Squares represent vertexes, arrows symbolize directed edges. Carbohydrate sequences may contain repeating units with non-stoichiometric modifications (STA), multiple connections between vertexes, cyclic subgraphs, alternative residue declarations and fuzzily defined capping unit locations. (B) Legend

1.2 Short historical review of carbohydrate sequence formats and their storage capabilities

An intuitive way of storing carbohydrate sequence topologies is a simple two dimensional sketch analogous to figure 2. This format, an ASCII 2-D plot, in close resemblance to IUPAC recommendations, was used by the first initiative for carbohydrate sequences storage, the Complex Carbohydrate Sequence Database (CCSD or often referred to as CarbBank). Subsequent initiatives using relational databases stored the carbohydrate sequences as strings, similar to protein or DNA - databases. These strings were obtained by an ordered traversal of the carbohydrate graphs and could thus serve as primary keys in database systems (canonicalized string representations such as LINUCS, GlyoSuite, LinearCode or BCSDB encoding). A newer approach to store the connectivity information in carbohydrate graphs by connection table – like representations (KCF, GlycoCT) was first introduced by the KEGG iniative. These connection tables can be naturally expressed in XML encodings (Glyde, CabosML, GlycoCT{XML}). The different formats used in glycobioinformatics have different capabilities to store the complex information potentially present in carbohydrate sequences (Table 1).

	Multigraph	Cyclisation	Repeating Units	Capping unit under-determined	Non-stoichiometric modification	Alternative residues
CCSD	-	+	+	(+)	(+)	-
LINUCS	-	+	+	-	-	-
GlycoSuite	-	-	-	+	-	+
BCSDB	(+)	-	(+)	-	-	+
LinearCode	-	+	-	+	-	+
KCF	+	+	+	-	-	-
GlycoCT	**+**	**+**	**+**	**+**	**+**	**+**
CabosML	(+)	+	+	-	-	-
Glyde	+	+	+	+	+	-

Table 1: A comparison of the structural information storage capabilities of the different sequence formats used in glycobioinformatics. Compare to Figure 2 for a more detailed description of the terms used. Legend: + can be stored, (+) partial solution or not explicitly defined, (-) cannot be stored.

These limitations of the existing sequence formats motivated us to define GlycoCT. Apart from the limitations shown in the table we found other restrictions in the existing sequence formats, especially concerning the handling of incomplete linkage information.

1.3 GlycoCT: Key features

The GlycoCT{condensed} variant can serve as an **unique identifier** for any glycan structure, even in the case of ambiguity in the structural description. The GlycoCT{XML} variant facilitates structura data parsing and data exchange. A **canonical labelling** for both nodes (residues) and edges (linkages) is accomplished via a sorting algorithm. This is used for internal adressing purposes, but can be used as well by external applications.

The following structural features can be encoded with GlycoCT:

- linear and branched structures
- compositions
- partially known topologies
- ambiguities on the linkage level
- fuzzy connectivities of subbranches
- statistical distribution of subgraphs along a carbohydrate sequence
- repeating units as completly distinct subgraphs
- circular structures
- multiconnected residues
- alternative subgraphs

Foreseen extensions of the format allow for the storage of additional properties:

- Isotopic labelling
- Structural description of fragmentation processes of oligosaccharides for mass spectrometric applications
- Peptides, proteins, organic small molecules or other substance classes can be attached to the sugar sequences

Basically, the GlycoCT format is a superset of the potential of all known sequence formats in glycobioinformatics with a number of enlargements for structurally underdetermined sequences, which are caused by biological heterogeneity and analytical limitations during structure elucidation. The graph oriented approach in conjunction with the canonical labelling will make future additions easy to implement.

Another major improvement over the existing formats used in glycobioinformatics is the introduction of a consistent naming scheme for the main basic components, the monosaccharides. A **controlled vocabulary** is of crucial importance for long-term stability of database projects. This naming scheme can be easily maintained with automatic checking routines, as it is fully machine readable and writable.

1.4 GlycoCT: Introduction

The format contains several discrete sections. A main graph and facultative subgraphs connected to it are modelled. It can be extended in the future by more sections to cover additional structural features and to hold more information if needed. The main sections are:

RES	List of connected entities. Mandatory. *(Residue)*
LIN	Topology information. A list of all topologically unique linkages *(Linkage)*
REP	Definition of repeating units as distinct subgraphs (*Repeat*)
ALT	Alternative subgraphs
UND	Underdetermined subgraphs. Attachment position of subgraph is unknown or non-stoichiometric modification.

Uppercase letters are mandatory for the section identifiers. The main RES-section is mandatory, followed by the optional LIN section. The rest of the sections appears in the order as shown above. Sequences start with the reducing terminus in general.

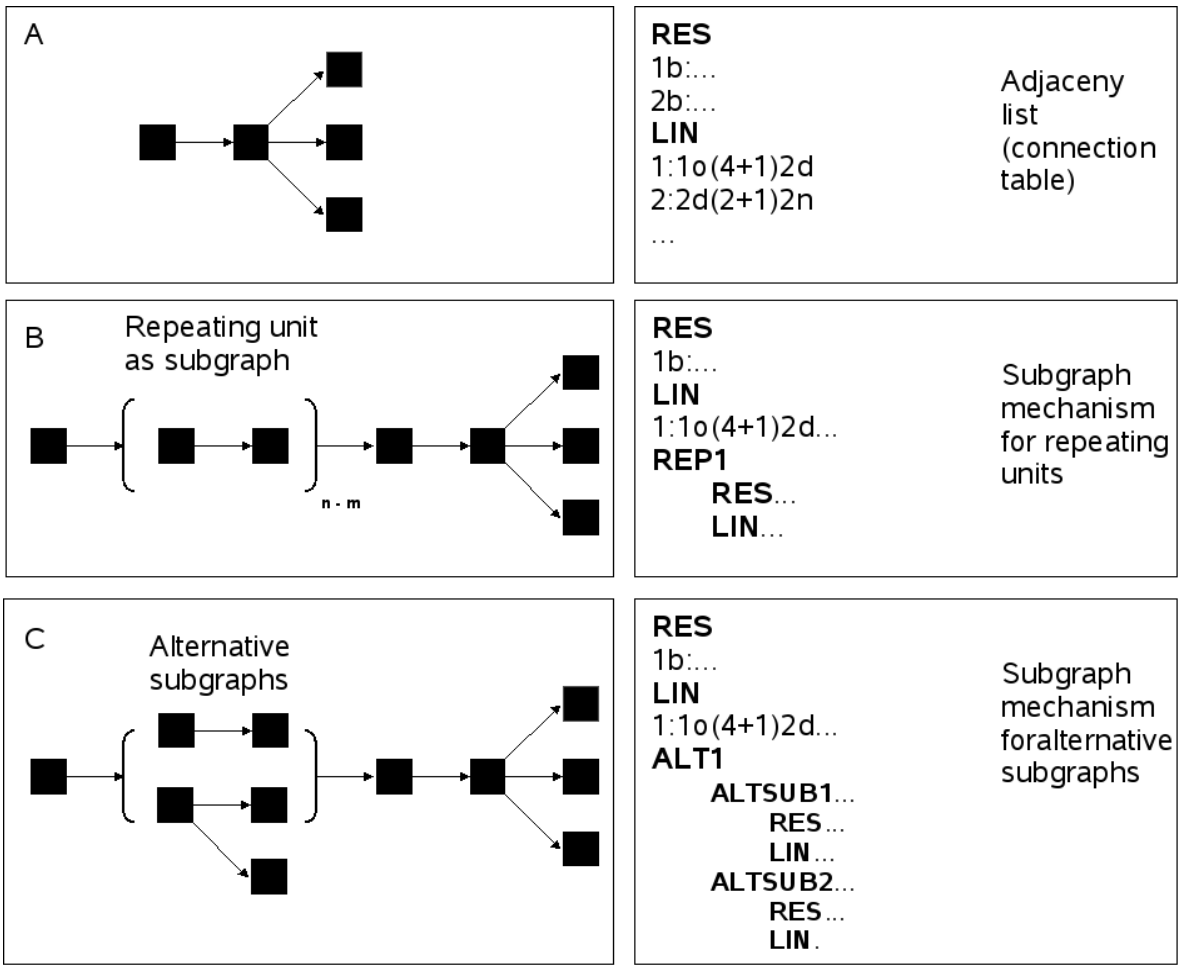


Figure 3: Main concepts of GlycoCT. (A) Tree - like structure are modelled via a connection table approach. (B) (C) Special feaures like repeat units or alternative residues or bigger alternative substructures are encoded as distinct subgraphs.

Figure 4 gives a more detailed view on the main RES- and LIN-sections.

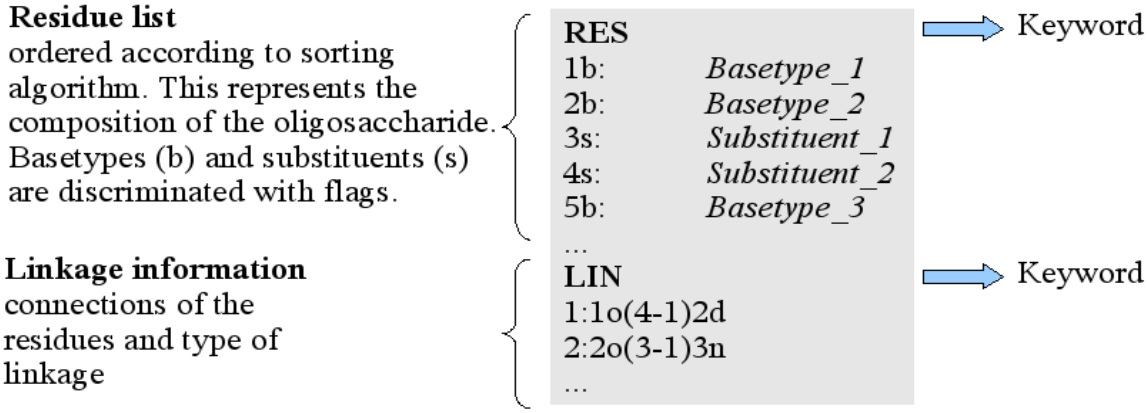


Figure 4: A first view on the main residue and linkage lists in GlycoCT.

To give a first overview of the format, the encoding of the well known N-glycan core is shown in figure 5.

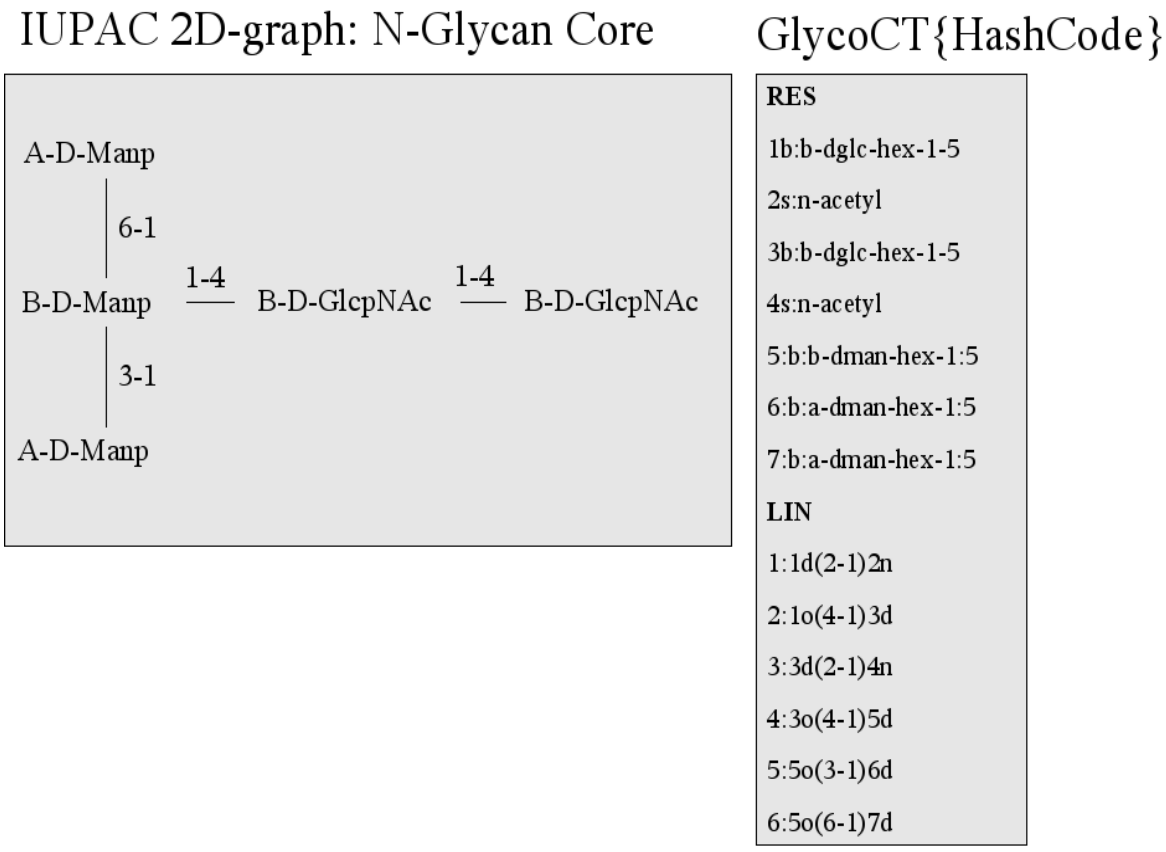


Figure 5: An example for the GlycoCT{condensed}. The main RES section contains the basetypes (b) and substituents (s), both derived from a controlled vocabulary. These entities are connected via the LIN section. Residues and linkages are both numbered canonically. Linkage type identifiers allow atomistic details of each linkage to be encoded.

2. GlycoCT{condensed}

This chapter describes the condensed form of GlycoCT, which is mainly designed to be a short hashcode for database applications and a human readable notation. GlycoCT{XML} is described in a subsequent chapter.

2.1 The RES section

This section contains all occuring basetypes, substituents and other entities in the current subgraph. The list is ordered according to the general sorting scheme (see respective chapter). The entries are numbered consecutively, followed by the mandatory type identifier, which has the following possible values:

b basetype
s substituent
n non-carbohydrate unit
r repeating unit
a alternative unit

The following sections explain the different types of residues used in GlycoCT.

2.1.1 Basetype naming conventions (b)

Definition:

A carbohydrate is a polyhydroxyaldehyde or – ketone. The substance class of alditols is treated as carbohydrates.

A basetype is the stereochemical skeletal structure of a carbohydrate without substituents.

This namespace is designed to result in unique identifiers for carbohydrates, including common modifications of this substance class. It is designed to be easily machine-readable and to resemble traditional IUPAC definitions as much as possible. Trivial IUPAC – names for carbohydrates are not allowed.

2.1.1.1 Subinformation: Superclass

Each carbohydrate descriptor is suffixed with its superclass, which defines the number of C-atoms in the main chain of the carbohydrate as defined by IUPAC. Figure 6 shows a table with the three letter codes and full names up to *Hexadecose* level.

Superclasses with more than 10 C-atoms can be named with the S[NN]-notation, theoretically allowing for basetypes with up to 99 C-atoms to be encoded.

number of linear oriented C-atoms	3-letter-code	long name
3	**TRI**	Triose
4	**TET**	Tetrose
5	**PEN**	Pentose
6	**HEX**	Hexose
7	**HEP**	Heptose
8	**OCT**	Octose
9	**NON**	Nonose
10	**DEC**	Decose
11	**S11**	Undecose
12	**S12**	Dodecose
13	**S13**	Tridecose
14	**S14**	Tetradecose
15	**S15**	Pentadecose
16	**S16**	Hexadecose

Figure 6: Three letter code for carbohydrate superclasses.

2.1.1.2 Subinformation: Carbohydrate stem types

A basic carbohydrate stem type is defined by its stereochemistry. The following basic stereochemically distinct entities are defined by IUPAC and will be used in the sequence format. These aldoses are the fundamentum of the namespace, each alteration from these basetypes is encoded in the carbohydrate nomenclature.

Configuration	3-letter-code	long name	superclass
D	**GRO**	Glyceraldehyde	TRI
D	**ERY**	Erythrose	TET
D	**RIB**	Ribose	PEN
D	**ARA**	Arabinose	PEN
D	**ALL**	Allose	HEX
D	**ALT**	Altrose	HEX
D	**GLC**	Glucose	HEX
D	**MAN**	Mannose	HEX
D	**TRE**	Threose	TET
D	**XYL**	Xylose	PEN
D	**LYX**	Lyxose	PEN
D	**GUL**	Gulose	HEX
D	**IDO**	Idose	HEX
D	**GAL**	Galactose	HEX
D	**TAL**	Talose	HEX
L	**GRO**	Glyceraldehyde	TRI
L	**ERY**	Erythrose	TET
L	**RIB**	Ribose	PEN
L	**ARA**	Arabinose	PEN
L	**ALL**	Allose	HEX
L	**ALT**	Altrose	HEX
L	**GLC**	Glucose	HEX
L	**MAN**	Mannose	HEX
L	**TRE**	Threose	TET
L	**XYL**	Xylose	PEN
L	**LYX**	Lyxose	PEN
L	**GUL**	Gulose	HEX
L	**IDO**	Idose	HEX
L	**GAL**	Galactose	HEX
L	**TAL**	Talose	HEX

Table 7: Stem names are chosen according to IUPAC. Additional three letter codes for missing stem types not defined by IUPAC have been added.

2.1.1.3 Carbohydrates with more than 4 stereogenic C-atoms

For carbohydrates with more than 4 stereogenic centers an additional rule is required. We follow established IUPAC conventions for the naming purposes (*IUPAC Nomenclature of carbohydrates, 2-Carb-2.2.2. Choice of parent name, http://www.chem.qmul.ac.uk/iupac/2carb/*), which result in composite names. A carbohydrate containing more than four chiral centres is named by adding two or more configurational prefixes to the stem name. Prefixes are assigned in order to the chiral centers in groups of four, beginning with the group proximal to C-1. The prefix relating to the group of carbon atom(s) farthest from C-1 (which may contain less than four atoms) is cited first.

See Figure 8 for an example of how to compose the stem names to name carbohydrates with more than 4 stereogenic C-atoms.

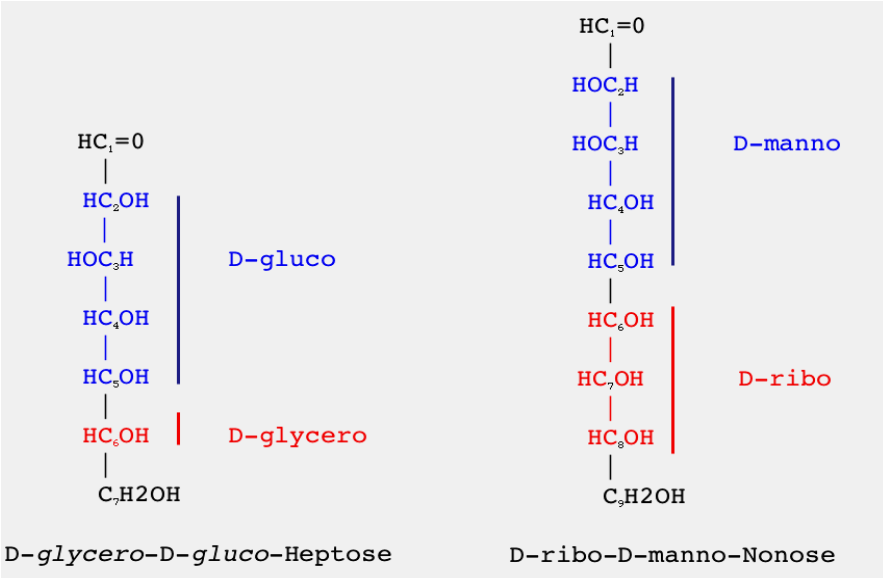


Figure 8: IUPAC defines a consistent naming scheme for carbohydrates with more than 4 stereocenters, resulting in composite names.

2.1.1.4 Subinformation: Anomeric center

This information is mandatory. Possible values:

a	alpha
b	beta
x	unknown
o	open = linear (no anomeric center exists)

2.1.1.5 Subinformation: Configuration

This information is mandatory. Possible values:

d	Dexter
l	Laevus
x	unknown

2.1.1.6 Subinformation: Ringsize

The descriptor for the ringsize follows always directly the basetype and is mandatory for all monomers. The position identifiers of the C-atoms forming a hemiacetal are written in numerical order and are seperated by a „:“.
In the case of unknown ringsize and -closure the special character x encodes the ambiguity. Linear structures receive a symbolic „0:0“ ([zero]:[zero]). These descriptors are mandatory.

2.1.1.7 Subinformation: Modifications of the basetypes altering stereoinformation

Modifications are noted in conjunction with the stem type in the following cases, as the stereoconfiguration is altered. This information should be stored in close proximity to the basetype.

- stereoloss by deoxigenation
- carbonyl function other than C1
- reduction of primary aldehyde function (alditols)
- double bond
- acidic function
- geminal OH
- *SP2*-hybrids
- *SP*-hybrids

Stereoloss by deoxigenation
Stereoloss is a common result of modifications of basetypes and is indicated by a „d“ (deoxy). We follow IUPAC-conventions for naming. For the definition of a basetype with stereoloss only the remaining stereogenic centers are taken into account, which results in composite names. Trivial names like Fucose, Rhamnose or Paratose are not allowed.

Carbonyl function other than C1
The keyword „keto“ is used to define the position of a carbonyl function at a position different from C1. It substitutes the original carbonyl function. Keto and alditol definitions on the same C-atom are not allowed.

Alditols
Reduction of the C1 aldehyde function is indicated by the keyword „aldi“. Only C1 reductions of basetypes are allowed. IUPAC recommendations for the correct naming of reduced aldoses are followed.

Double bond
The keywords „en“ indicates the existence of a double bond, joining two adjacent C-atoms in the backbone. Concurrent eliminations of OH – groups have to be indicated with the deoxigenation-flag.

An unknown deoxigenation pattern can be indicated with the modification „enx“.

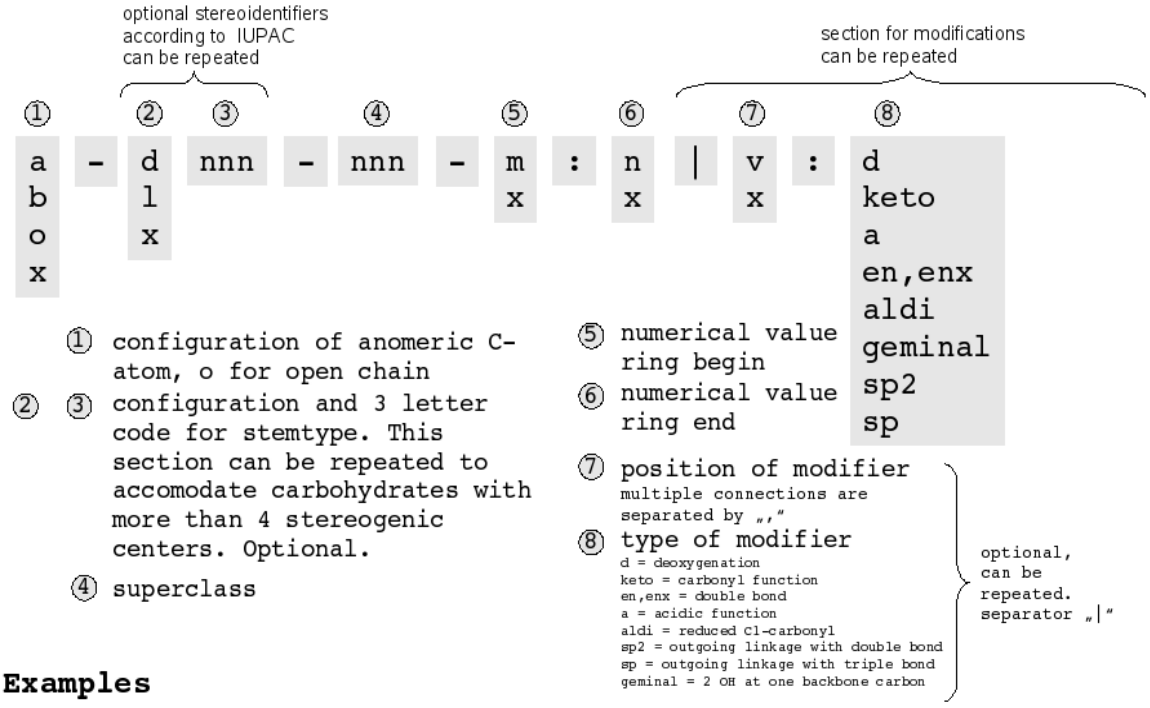
Acidic function
The keyword „a“ indicates the existence of an acidic function.

SP_2-Hybrids
SP2 -hybridisation due to a outgoing double bond to a substituent is indicated with the keyword „sp2“.

SP-Hybrids
SP -hybridisation due to a triple bond on terminal C-atoms is indicated with the keyword „sp“.

Geminal
Should a geminal substitution occur in the backbone of the basetype, it can be indicated with the „geminal“ keyword.

2.1.1.8 Summarizing schema for basetype descriptors



Examples

b-dglc-hex-1:5	β-D-Glucose, pyranose form
x-lgro-tet-0:0|3:d	?-3-deoxy-l-grotet (CarbBank notation)
a-drib-hex-x:x|3:d|6:d	Paratose: 3,6-dideoxy-D-ribo-hexose, alpha anomer
a-lgal-hex-1:5|6:d	LFuc, alpha anomer (KEGG notation)
b-dgro-dgal-hep-1:5|7:d	7-deoxy-β-D-Gro-D-Gal-Hepp

Figure 9: General schema for basetype naming.

2.1.2 Substituents (s)

The following table is the comprehensive list of common substituents for GlycoCT and has been derived from a thorough analysis of existing glycodatabase entries. While adding substituents to basetypes, as a general rule, basetype integrity has to be preserved as much as possible. Other chemical entities, which are not listed here, are attached to the sugar graph consequently as non-monosaccharide residues.

Monovalent substituents

Symbol	Formula	Bond order	Remarks
acetyl	**C**OCH3	1	
bromo	**Br**	1	
chloro	**Cl**	1	
ethyl	**C**H2CH3	1	
ethanolamine	CH2**N**HCH2OH	1	
flouro	**F**	1	
formyl	**C**HO	1	
glycolyl	**C**OCH2OH	1	
hydroxymethyl	**C**H2OH	1	
imino	**N**H	2	
iodo	**I**	1	
(r)-lactate	CH3**C**HCOOH	1	
(s)-lactate	CH3**C**HCOOH	1	
methyl	**C**H3	1	
n	**N**H2	1	amino
n-acetyl	**N**HCOCH3	1	aminoacetyl
n-alanine	**N**HCOCHNH2CH3	1	aminoalanine
n-dimethyl	**N**(CH3)2	1	aminoformyl
n-formyl	**N**HCHO	1	
n-glycolyl	**N**COCH2OH	1	
n-methyl	**N**HCH3	1	
n-succinate	**N**COCH2CH2COOH	1	
n-sulfate	**N**HSO3H	1	
n-triflouroacetyl	**N**HCOCF3	1	
nitrat	**N**O2	1	
phospate	**P**O3H2	1	
pyruvate	**C**OCOCH3	1	
sulfate	**S**O3H	1	
thio	**S**H	1	

Divalent substituents

(r)-pyruvate	CH2**C**COOH	2 * 1	
(s)-pyruvate	CH2**C**COOH	2 * 1	
(r)-lactate	CH3**CHC**O	2 * 1	
(s)-lactate	CH3**CHC**O	2 * 1	
anhydro	-H2O from basetye (intramolecular ether)	2 * 1	
lactone	-H2O from basetye (intramolecular ester)	2 * 1	
epoxy	-H20 from basetype (neighbouring C-atoms)	2 * 1	

Figure 10: A small, standarized substituent table is used as a seed list for GlycoCT. Future addtions to this list are welcome on a „as needed“ basis for common substitutions. Attachment positions are highlighted.

2.1.3 Non-monosaccharide entities (n)

Non-monosaccharide entities are marked in the residue list with the type identifier 'n' and a canonical number. They are further broken down in the non-monosaccharide section (NON). Non-monosaccharide entities can be optionally included in the GlycoCT format. Due to namespace inconsistencies (freetext identifiers for historical data) this may lead to a loss of the unique encoding. It may be advisable for certain database applications to store non-monosaccharide information in specialized data structures, separating glyco- and non-glyco data. Regardless, the following four subtypes for non-monosaccharide entities have been foreseen, but beware, this area is most probably subject to change in the future. Any GlycoCT – exporter program should as a minimum have a switch to suppress the output of non-monosaccharide entities.

2.1.3.1 Subtype: Historical entity

For backward compability with existing databases a freetext identifier for non-monosaccharide entities is put here. Beware of loss of uniqueness caused by synonyms.

2.1.3.2 Subtype: Small molecule

Uses an INCHI code to identify the non-monosaccharide entity uniquely.

2.1.3.3 Subtype: Peptide

Contains the full peptide sequence and an attachment postition.

2.1.3.4 Subtype: Protein

Stores a reference to an external protein sequence database, with database identifier, external protein ID, chain identifier and number of residue.

2.1.4 Subgraphs: Alternative Units (a)

Alternative units are marked in the residue list with the type identifier 'a' and a canonical number. They are further broken down in the alternative unit section (ALT).

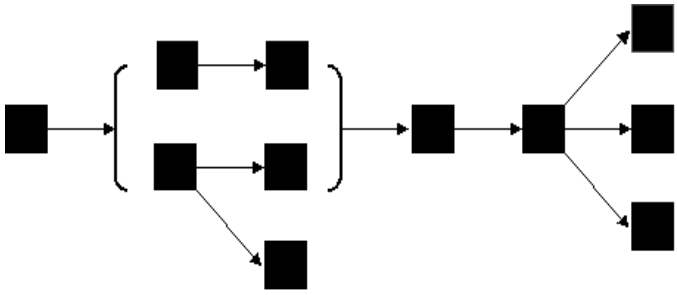


Figure 11: Alternative subgraphs are resolved in a specialized section.

2.1.5 Subgraphs: Repeat Units (r)

Repeat units are marked in the residue list with the type identifier 'r' and a canonical number.They are further broken down in the alternative unit section (REP).

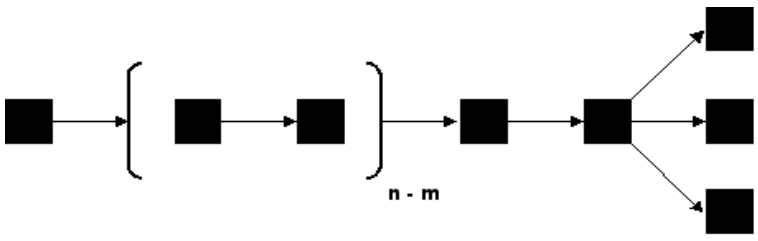


Figure 12: Repeat units are handled in a specialized section.

2.2 The LIN section

The LIN section contains the topological connectivities (linkages) of the entities in the RES section. Each connectivity is numbered consecutively. The list of connectivities is ordered as described in the chapter „Sorting GlycoCT“.

All linkages contain a **type identifier**, which indicates the substitution pattern:

o	hydrogen from OH – function removed and substituent attached at this position
h	hydrogen removed and substituent attached at this position
d	OH-function removed and substituted
n	linkage to substituents, non-monosaccharide entities, repeat or statistical units
x	unknown linkage type
r	prochiral H-atom removed, resulting in R-configuration
s	prochiral H-atom removed, resulting in S-configuration

The following rules should be followed when constructing the linkage information:
Basetype structure integrity should be preserved. Basetype-basetype linkages are always written with anomeric C receiving a deoxigenation. If a basetype-basetype linkages is achieved by two anomeric C atoms, the C-atom which comes at second place when following the preferred direction is deoxigenated. If no anomeric C-atom is involved in a basetype-basetype linkage, the C-atom which comes at second place when following the preferred direction is deoxigenated. If deoxigenation and substitution of backbone hydrogen happen on the same backbone C-atom, the basetype with the lower alphanumerical precedence is used.

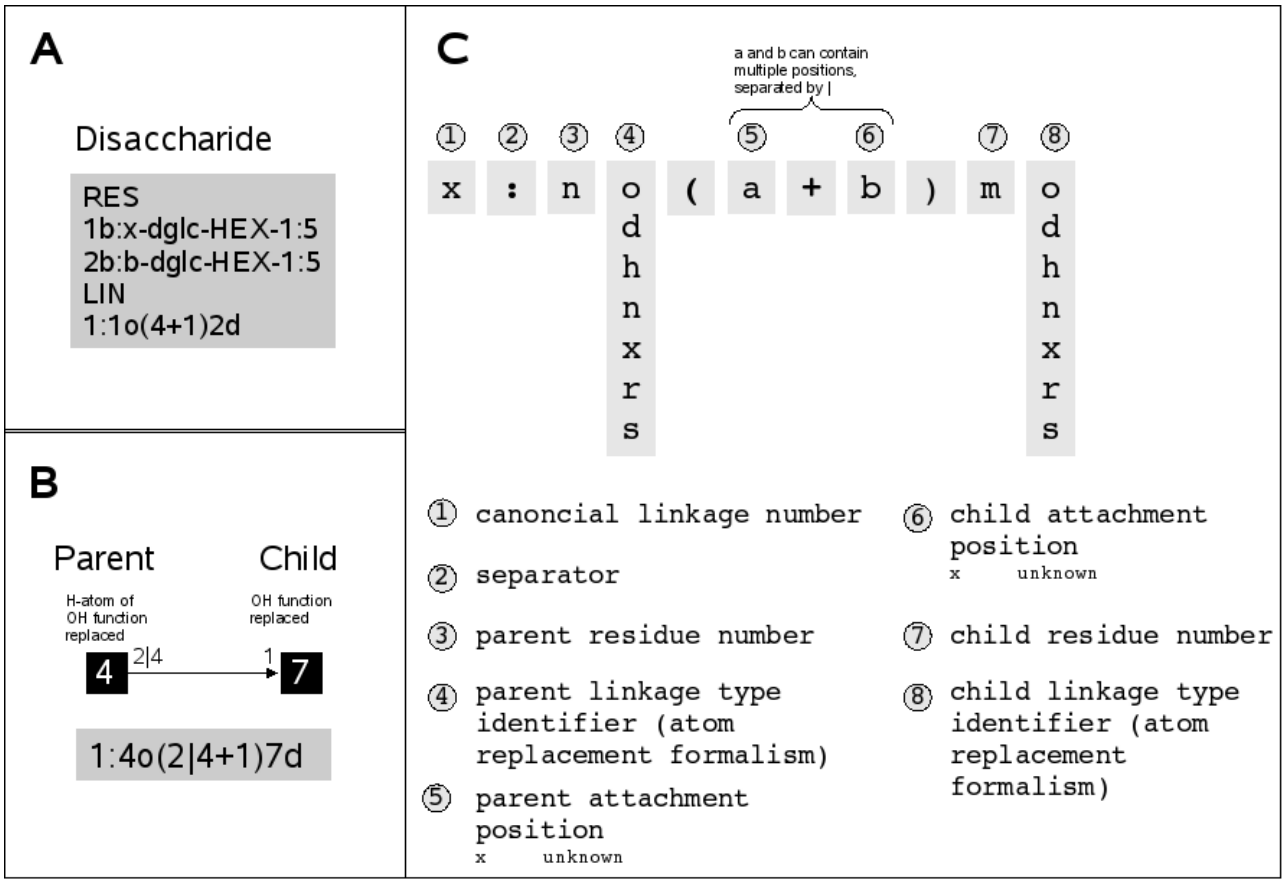


Figure 13: (A) A small disaccaride fragment of 2 glucoses in pyranose form linked via 4-1. (B) The linkage concept relies on a parent - child relationship of numbered residues. An atom replacement mechanism within each linkage results in a gateway to atomistic bonding details. (C) The linkage in GlycoCT, a schematical explaination.

2.2.1 Compositions

Carbohydrate compositions can be simply stored by listing the residues:

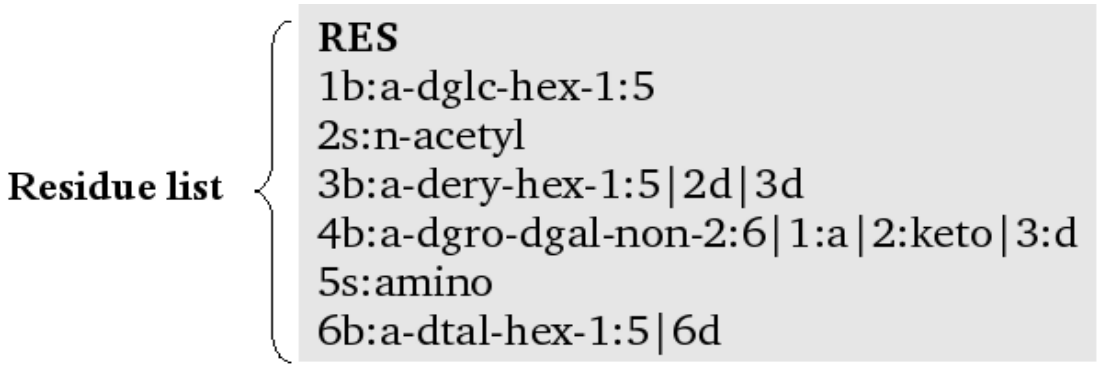


Figure 14: A composition stored in GlycoCT{condensed}.

2.2.2 Fragmented carbohydrate sequences

Graphs, for which only partial connectivity information is known, can be handled by the GlycoCT{condensed} as well by listing the known linkages in the LIN - section:

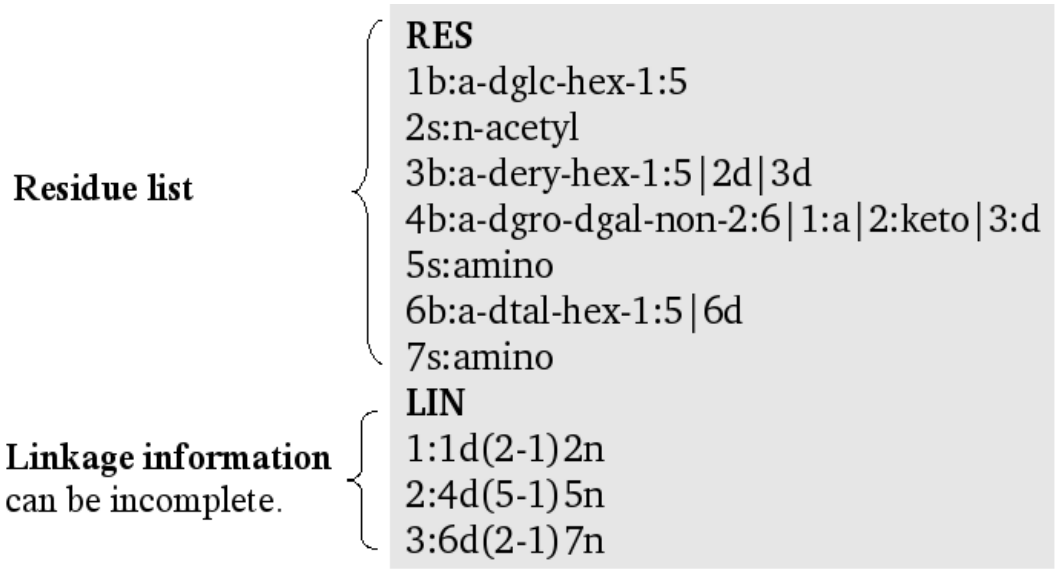


Figure 15: A fragmented carbohydrate sequence.Not all residues are connected to each other.

2.2.3 Undefined and partially known linkages

Different experimentally derived alternative linkages can be listed in the LIN – section. Should the attachment positions be totally unknown, a 'x' must indicate this.

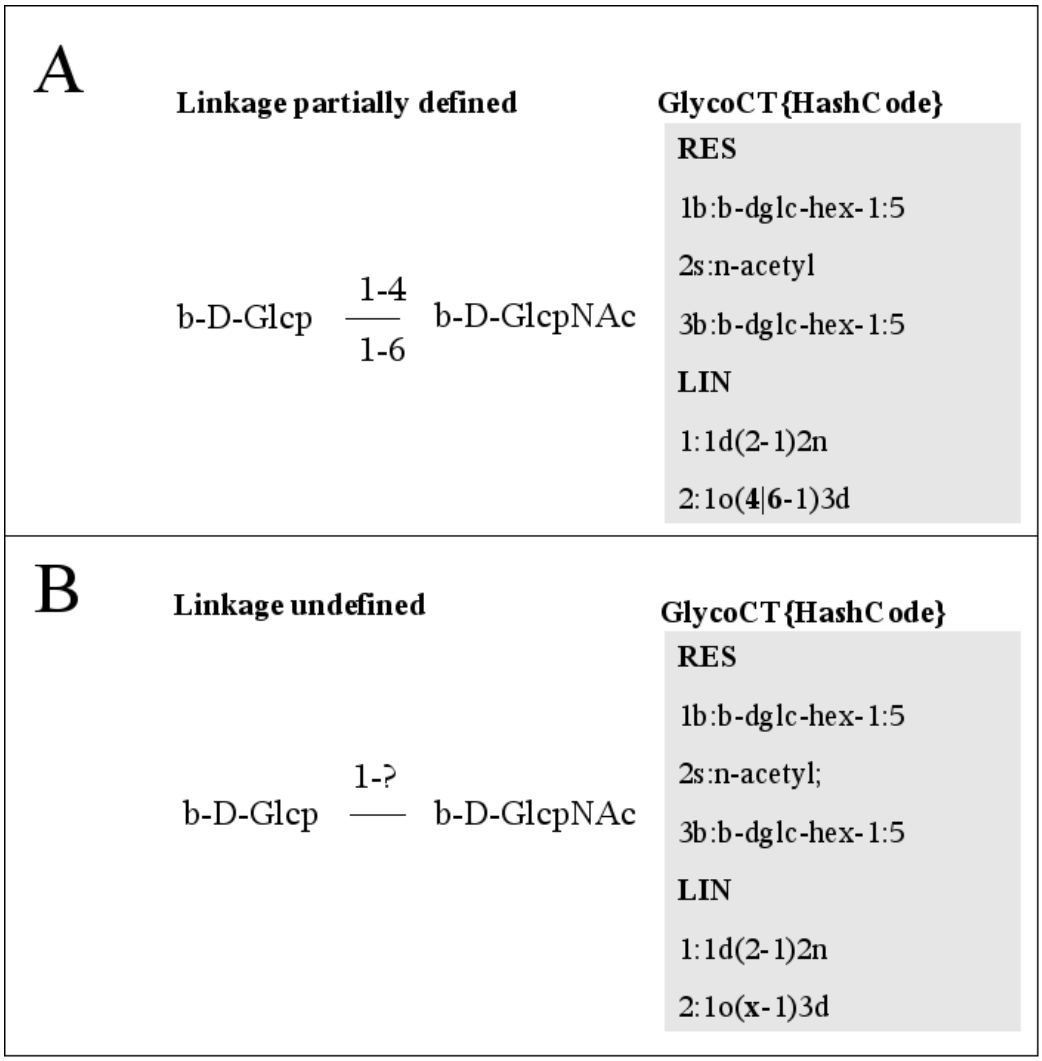
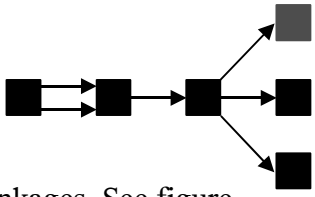


Figure 16: (A) Partly defined attachment positions can be enumerated with the character '|'. (B) Totally undefined linkages receive a 'x'.

2.2.4 Multiconnected residues - Multigraphs



The format can handle multiply connected residues by adding additional linkages. See figure 17.

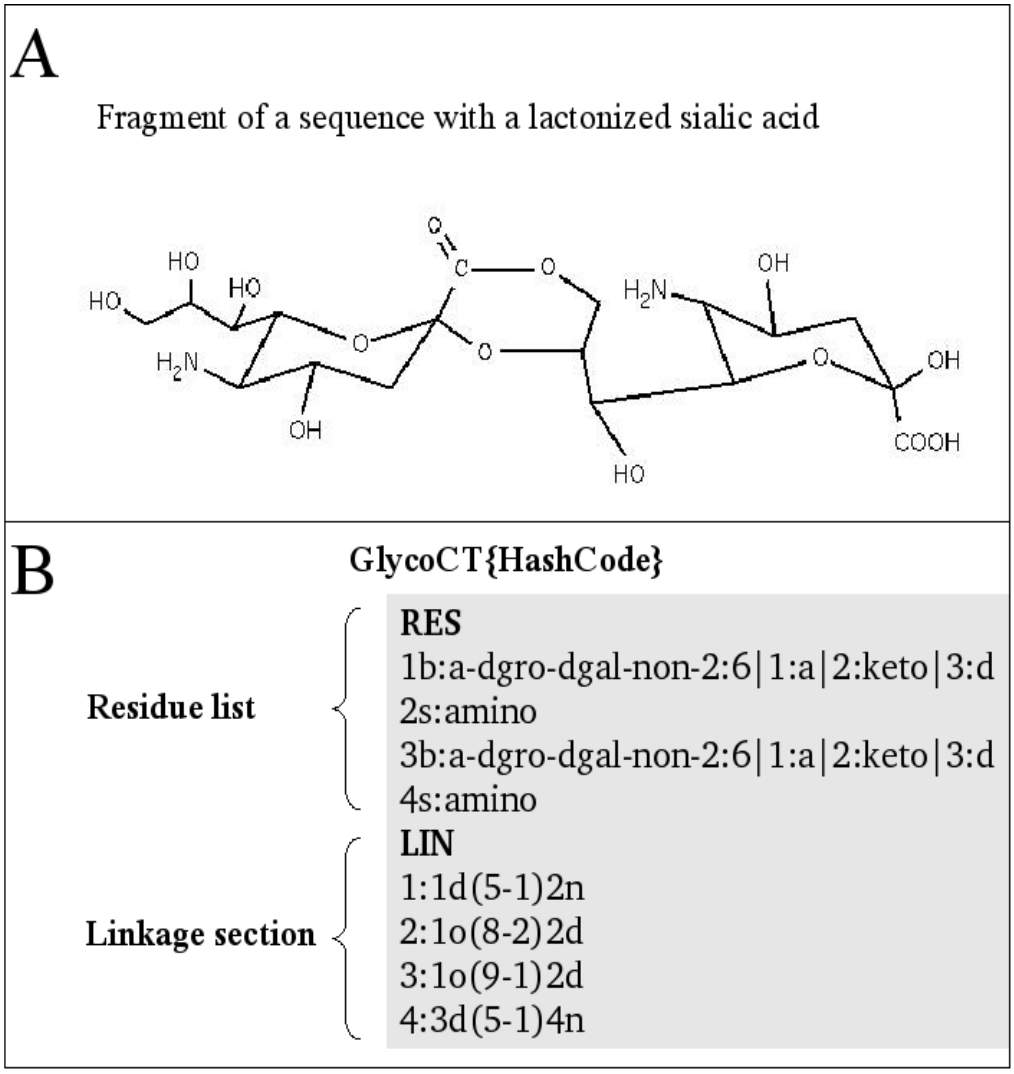
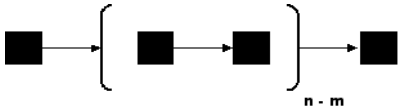


Figure 17: (A) Lactonization of two sialic acids. (B) Encoding of multiple connectivies between entities in the RES-section.

2.2.5 Circular sequences

Circular carbohydrate sequences can be written by adding approbiate linkages, which simply lead to cyclisation. The implications for the sorting algorithm are described in the respective chapter.

2.3 The REP section



Naturally occuring glycan structures especially from unicellular organisms or plants contain structural elements which are repeated n to m times in the resulting oligosaccharide. The section REP contains the repeating units as subgraphs with residues and connecting linkages which form the repeat unit. The multitude of the repeat unit needs to be noted with a starting „=“ and two integer numbers divided by a „-“. If the number of repeating units is known exactly, both numbers are the same. If it is unknown, a '-1' is used to indicate it.

More than one repeat section may exist in a complex sugar graph. All repeating units are listed in the sequence of the general sorting scheme with consecutiv numbers. The smallest possible repeat unit has to be declared.

The prerequisite for the usage of the repeating section is a number of at least **seven** repeating elements.

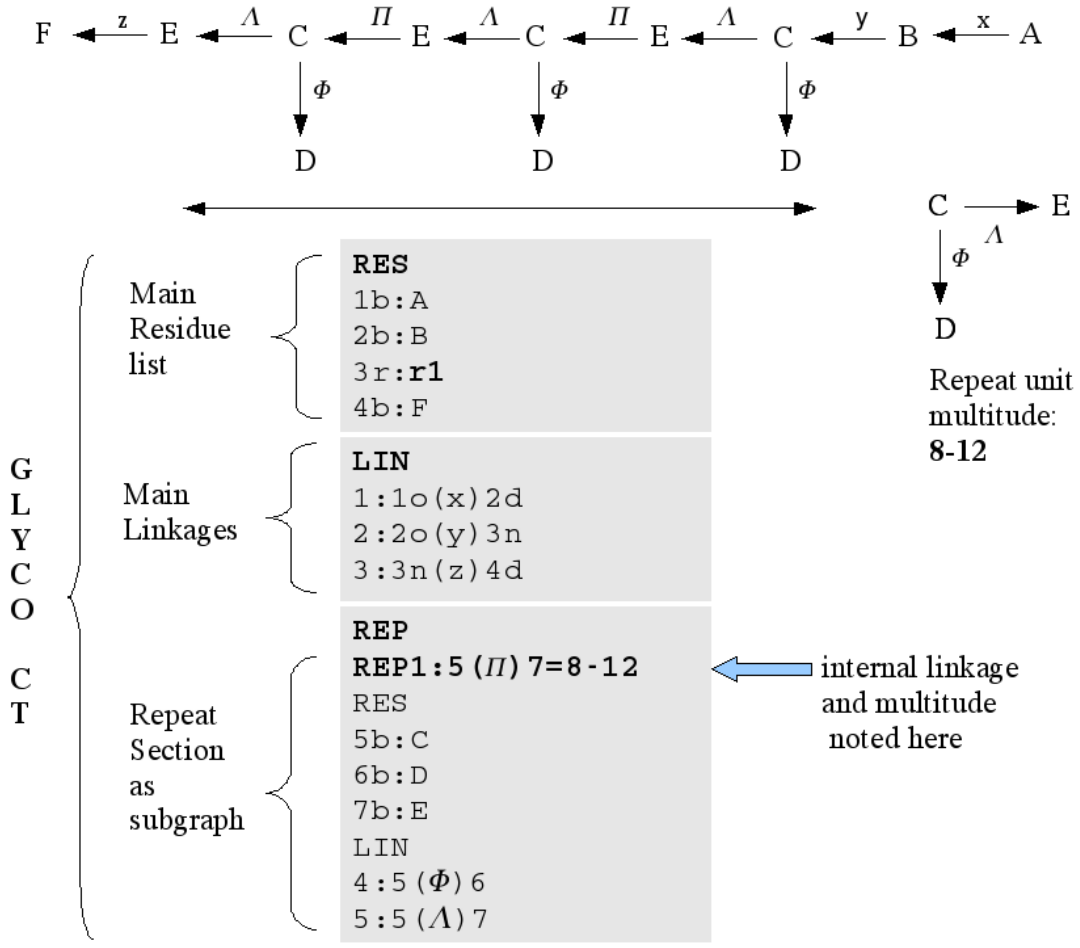


Figure 18: Schematic drawing. Repeating Units are represented as subgraphs.

Figure 19 shows an example of nested repeating units.

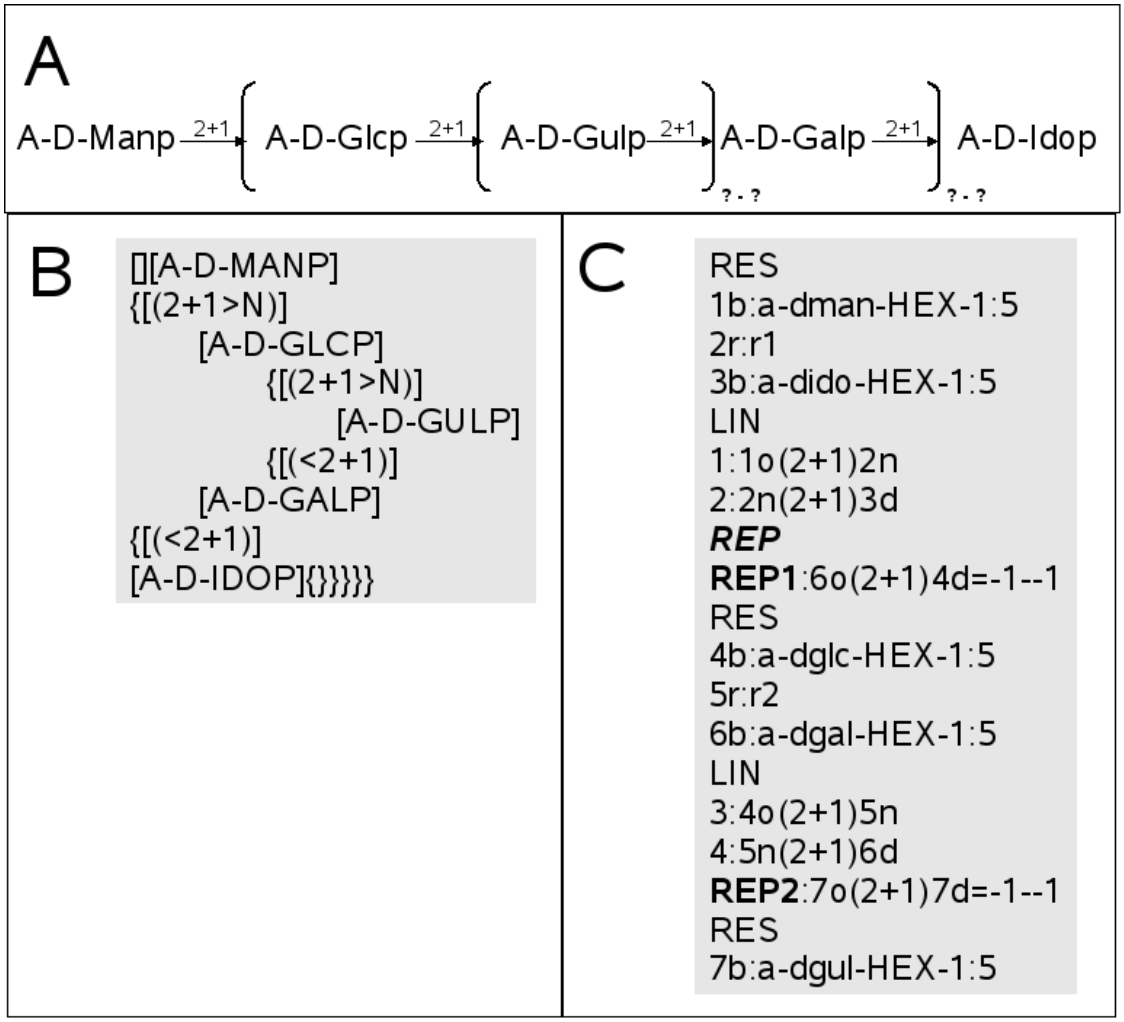
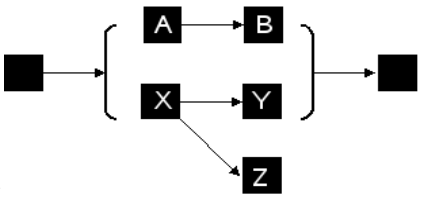


Figure 19: (A) A nested repeat.(B) The same structure in LINUCS-encoding. (C) This structure in GlycoCT{condensed}. The multitude of each repeating unit is unknown, indicated by a '-1'.

2.4 The ALT section



The ALT-section is designed to store alternative subgraphs within a carbohydrate sequence. It contains two or more subgraphs, in each of them one lead-in and one or more lead-out nodes are marked.

Rule: If the ALT-section can be substituted by normal enumerated linkage alternatives, then it should not be used. ALT-sections contain mandatorily a link to their parent node, which is noted with the following notation:

LEAD-IN RES: <canonical *number of parent node*>

If the alternative residues are non-terminal, child nodes are referred to with the following notation:

LEAD-OUT RES <canonical *number of parent node*> + <canonical *number of child node*>

Multiple LEAD-OUT identifiers are separated by a '|'

To guarentee uniqueness, the ALT – section should be as small as possible. Overlapping, common residues and linkages have to be taken out and written to the hierachically higher (super)graph. See figure 20 for examples.

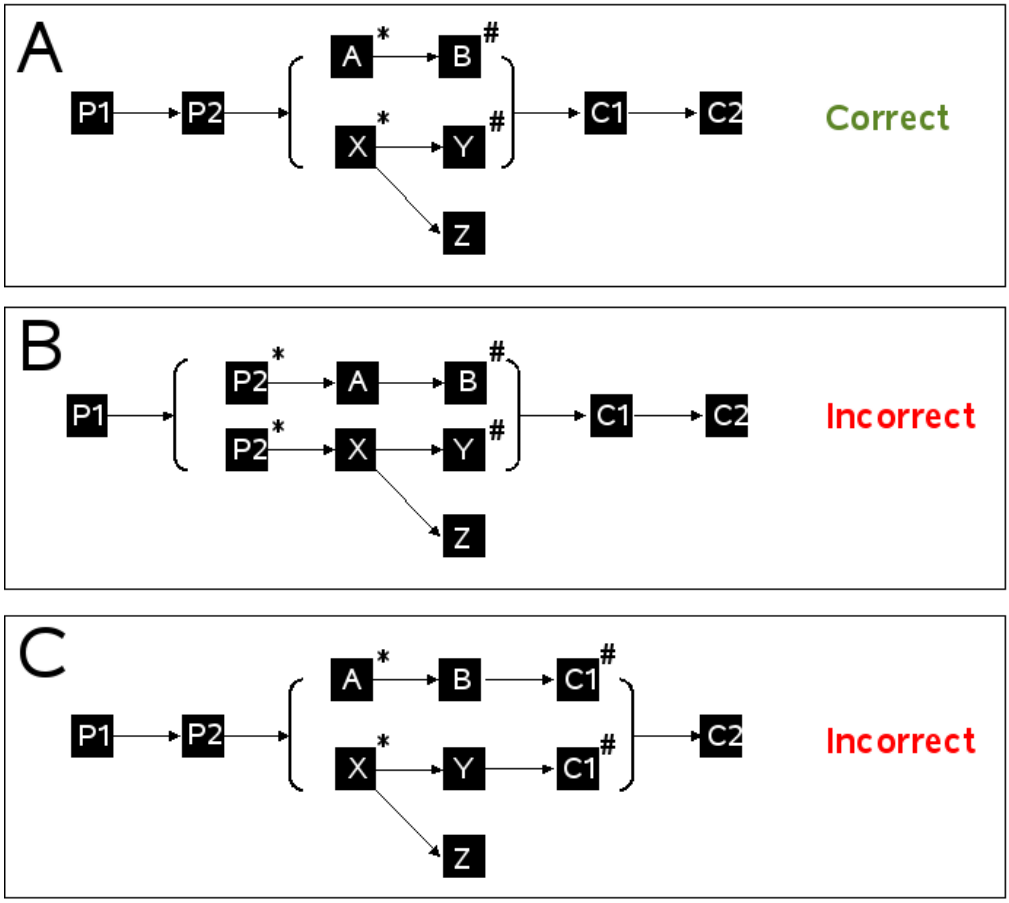
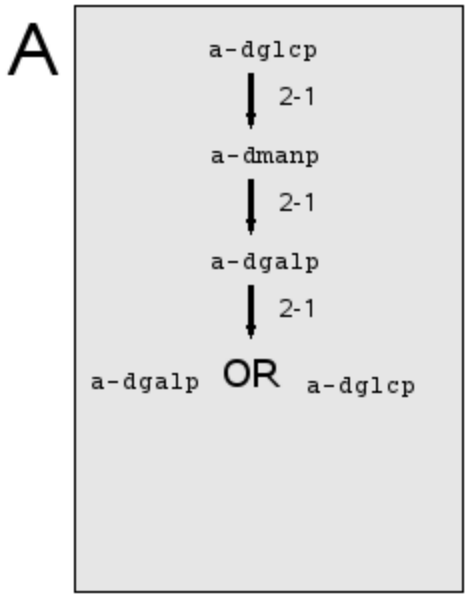


Figure 20: Alternative subgraphs. '' marks lead-in nodes, '#' for lead out nodes. A) Correct declaration. (B), (C) Subgraph declaration violates the optimisation rule. The graphs in (B) and (C) express the same information as (A).*

The following example shows a simple alternative residue declaration:

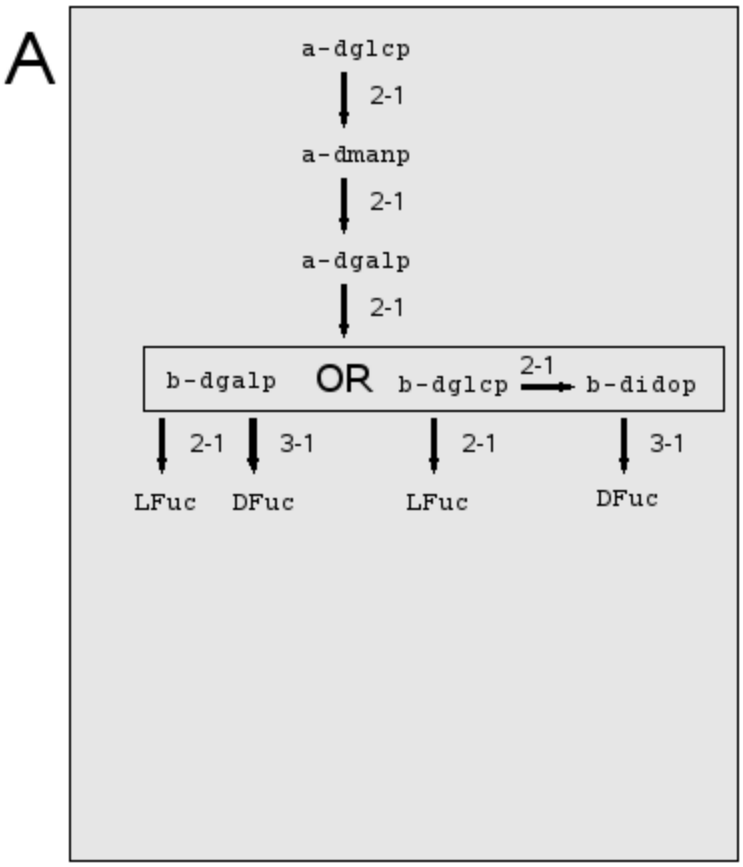


B

```
RES
1b:a-dglc-HEX-1:5
2b:a-dman-HEX-1:5
3b:a-dgal-HEX-1:5
4a:a1
LIN
1:1o(2+1)2d
2:2o(2+1)3d
3:3o(2+1)4d
ALT
ALT1
ALTSUBGRAPH1
LEAD-IN RES:5
RES
5b:b-dgal-HEX-1:5
ALTSUBGRAPH2
LEAD-IN RES:6
RES
6b:b-dglc-HEX-1:5
```

Figure 21: (A) Glucopyranose or Galactopyranose are terminal residues of this linear sequence. (B) GlycoCT{condensed} with alternative subgraphs.

A more complex GlycoCT{condensed} is shown in the next figure:

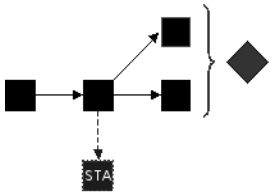


B

```
RES
1b:a-dglc-HEX-1:5
2b:a-dman-HEX-1:5
3b:a-dgal-HEX-1:5
4a:a1
5b:a-lgal-HEX-0:0|6:d
6b:a-dgal-HEX-0:0|6:d
LIN
1:1o(2+1)2d
2:2o(2+1)3d
3:3o(2+1)4d
4:4o(2+1)5d
5:4o(3+1)6d
ALT
ALT1
ALTSUBGRAPH1
LEAD-IN RES:7
LEAD-OUT RES:7+5|7+6|
RES
7b:b-dgal-HEX-1:5
ALTSUBGRAPH2
LEAD-IN RES:8
LEAD-OUT RES:8+5|9+6|
RES
8b:b-dglc-HEX-1:5
9b:b-dido-HEX-1:5
LIN
6:8o(2+1)9d
```

Figure 22: (A) An alternative subgraph is part of a bigger connected structure (B) Corresponding GlycoCT{condensed}. Special LEAD-IN and LEAD-OUT sections link the alternative subgraph to the main graph.

2.5 The UND section – concurrent information



The section UND (underdetermined structures) contains information about substructures which have a incomplete connectivity information to the other graph components. The UND – subgraphs carry a probability value, which can be used to model statistically distributed modifications of carbohydrate graphs. UND subgraphs are always terminal and numbered consecutivly.

As the UND section is concurrent to the main sections of GlycoCT it references one or more nodes as parents and is linked via the canonical node numbers to the main graph. The linkage information to the different possible parents is fixed, though, and cannot be switched for different parents.

This model can be used for two following cases:

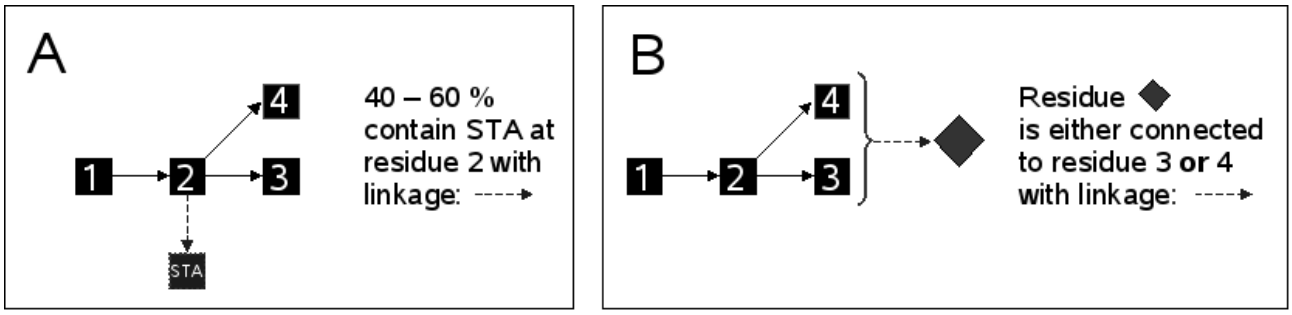
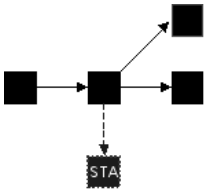


Figure 23: Usage of the UND section (A) Statistically distributed subgraphs. (B) Underdetermined capping unit location.

Each UND-subsection is an independent subgraph, which is linked to the main graph via the canoncial IDs of the parents. The linkage to the main graph is defined with the SubtreeLinkage information. The SubTreeLinkage information is indexed with an ID, its parent is defined via the ParentIDs. The SubTreeLinkage information points to the root node in the UND subsection, which is defined as the residue without parent.

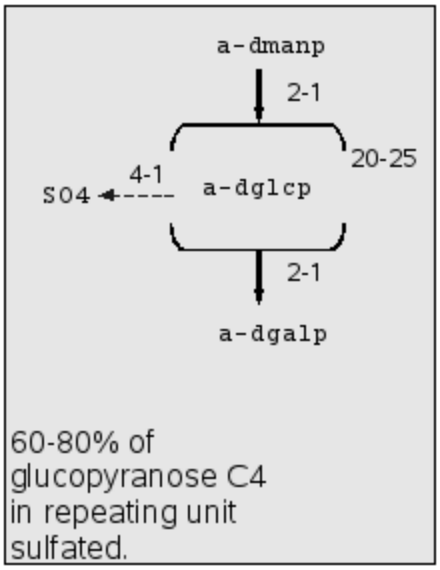
```
UND                                  Keyword for beginning of section
UND1:100.0:100.0                     ID of subsection and probabilities
ParentIDs:3|1                        Parent Node IDs in main graph
SubtreeLinkageID1:o(2+1)d            Linkage to Main Graph
RES                                  Residue List
...
LIN                                  Optional connectivity information
...
```

2.5.1 UND-section for statistically distributed subgraphs



The following figure shows an example of statistically distributed modification of a carbohydrate sequence, here a sulfate which is distributed over a repeating unit.

A

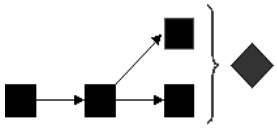


B

```
RES
1b:a-dman-HEX-1:5
2r:r1
3b:a-dgal-HEX-1:5
LIN
1:1o(2+1)2n
2:2n(2+1)3d
REP
REP1:4o(2+1)4d=20-25
RES
4b:a-dglc-HEX-1:5
UND
UND1:60.0:80.0
ParentIDs:4
SubtreeLinkageID1:o(4+1)n
RES
5s:sulfate
```

Figure 24: Statistical modifications modelled with the UND section. (A) The carbohydrate sequence contains a repeating unit, which is modified to a varying extent. (B) Resulting GlycoCT{condensed}. The UND - section is concurrent to the main graph. UND - subgraphs are numbered consecutivly and carry a minor and major probabilistic value. Via their ParentIDs and SubtreeLinkage the main graph is referenced. The root nodes (parentless vertexes) of each UND subgraph are connected to the main graph.

2.5.2 UND-sections for underdetermined capping unit location



The next two examples show usages of the terminal subgraph mechanism in GlycoCT:

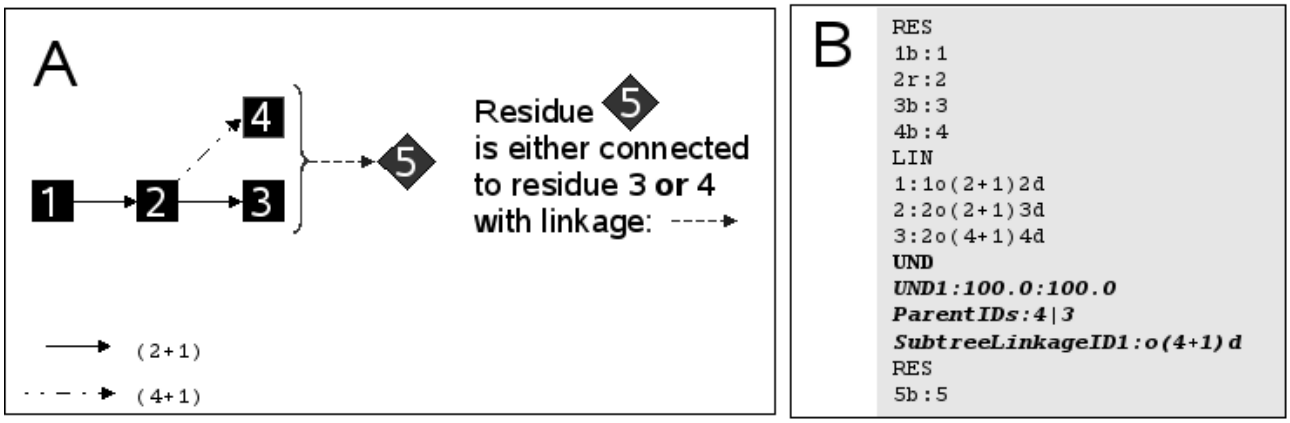


Figure 25: Terminal residue location unknown - schematic drawing. (A) A carbohydrate graph, the exact location of residue 5 is not reported. (B) Resulting GlycoCT{condensed}. UND subgraphs has here a probability of 100, but has two different ParentIDs.

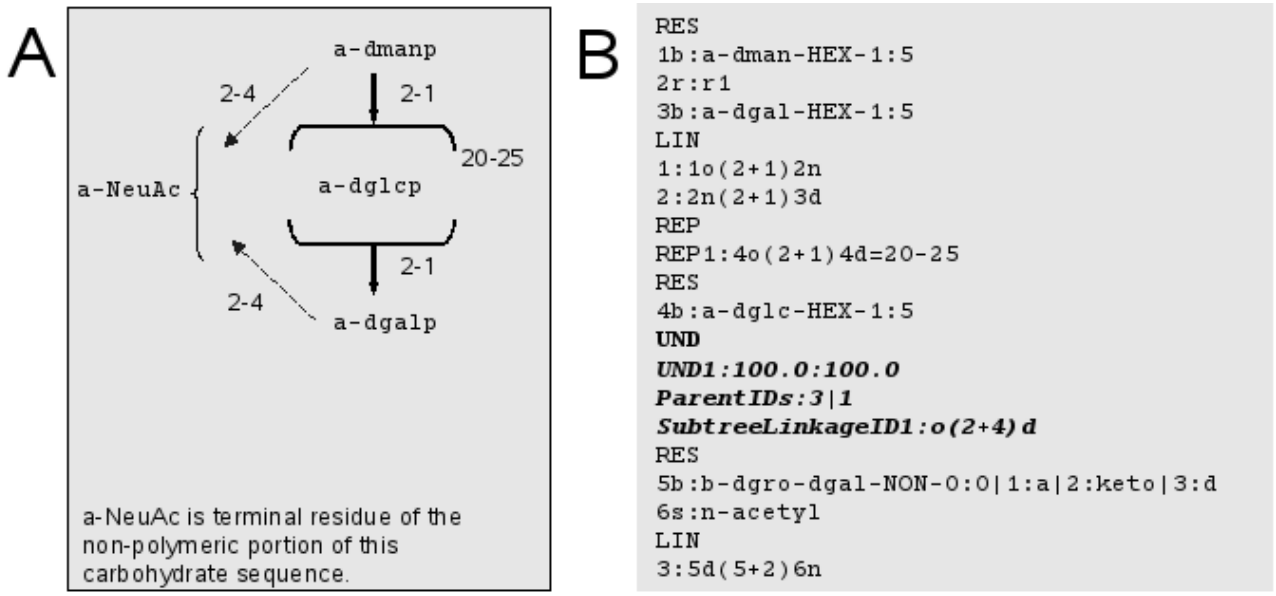


Figure 26: Terminal residue / subgraph attachment uncertainty. (A) A sialic acid is known to be linked to the oligomeric portion of a polysaccharide. (B) Resulting GlycoCT code. The UND section with a probability of 100 and ***two*** *parent IDs is used to model this structure.*

2.6 The falcultative ISO section

Certain experimental techniques require a substitution or enrichment of atom types. These are encoded in the ISO section, using the following abbreviations:

Name	Abbreviation
Deuterium	d
Carbon-13	c13
Nitrogen-15	n15
Nitrogen-14	n14
Oxygen-17	o17
Phosphor-31	p31

These substitution types are currently defined for basetypes:

c exchange of backbone carbon
o exchange of O
h exchange of H
r exchange of R-prochiral H
s exchange of S-prochiral H

Syntax:

[no]**:**[residue number][type identfier]**(**[position-[atom_type]**)**

The following example shows a C13-exchange on C6 of GlcNac:

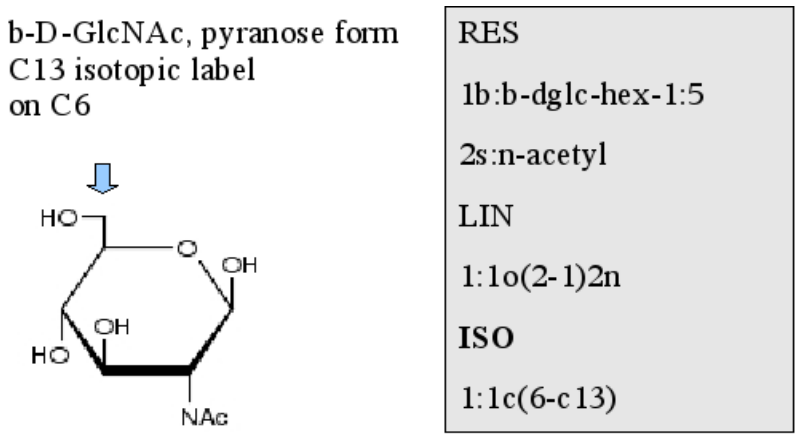


Figure 27: An extension of GlycoCT simplifies encoding of isotopic alterations.

2.7 The facultative NON section

It should be noted that applications in glycobioinformatics handling non-monosaccharide entities can externalize this information to different tables, so this information may not appear in GlycoCT. Nevertheless, handling of non-monosaccharide entities in GlycoCT is foreseen. Principally four classes of non-monosaccharides are to be distinguished, connected to the main graph via parent ID and linkage information:

Subtype: Historical entity
For backward compability with existing databases a freetext identifier for non-monosaccharide entities is put here. Beware of loss of uniqueness caused by synonyms.

Keyword:

HistoricalEntity: <NAME>

Subtype: Small molecule
Uses an INCHI code to identify the non-monosaccharide entity uniquely.

Keyword:

SmallMolecule: <INCHI>

Subtype: Peptide
Contains the full peptide sequence and an attachment postition.

Keyword:

Peptide: <SEQUENCE>

Subtype: Protein
Stores a reference to an external protein sequence database, with database identifier, external protein ID, chain identifier and number of residue.

PROTEIN

DB: <Identifier>

ID: <Identifier>

CHAIN: <Identifier>

RESIDUENUMBER: <Identifier>

These nonmonosaccharides can be linked via the canonical node numbers to any entity within GlycoCT.

Example: Mannopyranose with 5A-CHOLESTAN-1B,3B,16B,22S-TETROL

RES

1b:a-dman-HEX-1:5

NON1

Parent:1

Linkage:d(6-16)n

HistoricalEntitiy: 5A-CHOLESTAN-1B,3B,16B,22S-TETROL

3. GlycoCT{XML}

The schema for GlycoCT{XML} can be found in appendix 1. It uses the same main definitions and general terms as the GlycoCT{condensed}. The sorting algorithm is applied during the traversal for the XML document generation.

The following figure shows a small trisaccharide fragment:

A

```
RES
1b:a-dglc-HEX-1:5
2b:a-dman-HEX-1:5
3b:a-dgal-HEX-1:5
LIN
1:1o(2+1)2d
2:2o(2+1)3d
```

B

```
<?xml version="1.0" encoding="UTF-8"?>
<sugar version="1.0">
  <residues>
    <monosaccharide id="1" type="b" anomer="a" superclass="hex"
ringStart="1" ringEnd="5" name="a-dglc-HEX-1:5">
      <basetype id="1" type="dglc" />
    </monosaccharide>
    <monosaccharide id="2" type="b" anomer="a" superclass="hex"
ringStart="1" ringEnd="5" name="a-dman-HEX-1:5">
      <basetype id="1" type="dman" />
    </monosaccharide>
    <monosaccharide id="3" type="b" anomer="a" superclass="hex"
ringStart="1" ringEnd="5" name="a-dgal-HEX-1:5">
      <basetype id="1" type="dgal" />
    </monosaccharide>
  </residues>
  <linkages>
    <connection id="1" parent="1" child="2">
      <linkage id="1" parentType="o" childType="d">
        <parent pos="2" />
        <child pos="1" />
      </linkage>
    </connection>
    <connection id="2" parent="2" child="3">
      <linkage id="2" parentType="o" childType="d">
        <parent pos="2" />
        <child pos="1" />
      </linkage>
    </connection>
  </linkages>
</sugar>
```

Figure 28: GlycoCT{XML}. A small linear trisaccharide. (A) GlycoCT{condensed} (B) GlycoCT{XML}. The connection – element connects residues and is capable of storing multiple linkages.

Here you find a more complex example, with alternative and repeat units. Figure 29 shows the same structure in GlycoCT{condensed}. This structure contains a repeat unit and an alternative unit.

```
RES
1b:a-dman-HEX-1:5
2r:r1
3b:a-dgal-HEX-1:5
4a:a1
5b:a-lgal-HEX-0:0|6:d
6b:a-dgal-HEX-0:0|6:d
LIN
1:1o(2+1)2n
2:2n(2+1)3d
3:3u(2+1)4u
4:4u(2+1)5u
5:4u(3+1)6u
REP
REP1:7o(2+1)7d=-1--1
RES
7b:a-dglc-HEX-1:5
ALT
ALT1
ALTSUBGRAPH1
LEAD-IN RES:8
LEAD-OUT RES:8+5|8+6|
RES
8b:b-dgal-HEX-1:5
ALTSUBGRAPH2
LEAD-IN RES:9
LEAD-OUT RES:9+5|
10+6|
RES
9b:b-dglc-HEX-1:5
10b:b-dido-HEX-1:5
LIN
6:9u(2+1)10u
```

Figure 29: GlycoCT{condensed} of a more complex sequence with alternative units and repeating elements.

```
<?xml version="1.0" encoding="UTF-8"?>
<sugar version="1.0">
  <residues>
    <monosaccharide id="1" type="b" anomer="a" superclass="hex"
ringStart="1" ringEnd="5" name="a-dman-HEX-1:5">
      <basetype id="1" type="dman" />
    </monosaccharide>
    <repeat id="2" repeatId="1" />
    <monosaccharide id="3" type="b" anomer="a" superclass="hex"
ringStart="1" ringEnd="5" name="a-dgal-HEX-1:5">
      <basetype id="1" type="dgal" />
    </monosaccharide>
    <alternative id="4" alternativeId="1" />
    <monosaccharide id="5" type="b" anomer="a" superclass="hex"
ringStart="0" ringEnd="0" name="a-lgal-HEX-0:0|6:d">
      <basetype id="1" type="lgal" />
      <modification type="d" pos_one="6" />
    </monosaccharide>
    <monosaccharide id="6" type="b" anomer="a" superclass="hex"
ringStart="0" ringEnd="0" name="a-dgal-HEX-0:0|6:d">
      <basetype id="1" type="dgal" />
      <modification type="d" pos_one="6" />
    </monosaccharide>
  </residues>
  <linkages>
    <connection id="1" parent="1" child="2">
      <linkage id="1" parentType="o" childType="n">
        <parent pos="2" />
        <child pos="1" />
      </linkage>
    </connection>
    <connection id="2" parent="2" child="3">
      <linkage id="2" parentType="n" childType="d">
        <parent pos="2" />
        <child pos="1" />
      </linkage>
    </connection>
    <connection id="3" parent="3" child="4">
      <linkage id="3" parentType="x" childType="x">
        <parent pos="2" />
        <child pos="1" />
      </linkage>
    </connection>
    <connection id="4" parent="4" child="5">
      <linkage id="4" parentType="x" childType="x">
        <parent pos="2" />
        <child pos="1" />
      </linkage>
    </connection>
    <connection id="5" parent="4" child="6">
      <linkage id="5" parentType="x" childType="x">
        <parent pos="3" />
        <child pos="1" />
      </linkage>
    </connection>
  </linkages>
  <repeat>
    <unit id="1" minOccur="-1" maxOccur="-1">
      <residues>
        <monosaccharide id="7" type="b" anomer="a" superclass="hex" ringStart="1" ringEnd="5"
name="a-dglc-HEX-1:5">
          <basetype id="1" type="dglc" />
        </monosaccharide>
      </residues>
      <linkages />
```

```
      <internalLinkage parent="7" child="7">
        <linkage id="6" parentType="o" childType="d">
          <parent pos="2" />
          <child pos="1" />
        </linkage>
      </internalLinkage>
    </unit>
  </repeat>
<alternative>
    <unit id="1">
      <substructure>
        <residues>
          <monosaccharide id="11" type="b" anomer="b" superclass="hex" ringStart="1" ringEnd="5"
name="b-dgal-HEX-1:5">
            <basetype id="1" type="dgal" />
          </monosaccharide>
        </residues>
        <linkages />
        <lead_in residue_id="11" />
        <lead_out residue_id="11" connected_to="5" />
        <lead_out residue_id="11" connected_to="6" />
      </substructure>
      <substructure>
        <residues>
          <monosaccharide id="12" type="b" anomer="b" superclass="hex" ringStart="1" ringEnd="5"
name="b-dglc-HEX-1:5">
            <basetype id="1" type="dglc" />
          </monosaccharide>
          <monosaccharide id="13" type="b" anomer="b" superclass="hex" ringStart="1" ringEnd="5"
name="b-dido-HEX-1:5">
            <basetype id="1" type="dido" />
          </monosaccharide>
        </residues>
        <linkages>
          <connection id="7" parent="12" child="13">
            <linkage id="10" parentType="o" childType="d">
              <parent pos="2" />
              <child pos="1" />
            </linkage>
          </connection>
        </linkages>
        <lead_in residue_id="12" />
        <lead_out residue_id="12" connected_to="5" />
        <lead_out residue_id="13" connected_to="6" />
      </substructure>
    </unit>
  </alternative>
</sugar>
```

4. GlycoCT{compressed}

GlycoCT{condensed} can be compressed with the LZW-algorithm to yield GlycoCT{compressed}. The reduced storage space can be beneficial for database applications.

A

```
RES
1b:a-dman-HEX-1:5
2r:r1
3b:a-dido-HEX-1:5
LIN
1:1o(2+1)2n
2:2n(2+1)3d
REP
REP1:6o(2+1)4d=11-11
RES
4b:a-dglc-HEX-1:5
5r:r2
6b:a-dgal-HEX-1:5
LIN
3:4o(2+1)5n
4:5n(2+1)6d
REP2:7o(2+1)7d=10-10
RES
7b:a-dgul-HEX-1:5
```

B

```
H4sIAAAAAAAAAFWNywrCM
BBF9/MligR6Jy8YcBlQEJ
F24zYaEKGmUPD/rRmouBg
YzoVz+jQQbpJN
eeVqDulqIJ54lhlkG3+Wa
eWn45kgmDa8w5YrsXBtvy
3Up8v3IEFnV/aAARY4kGu
qx3hfVX5JMAXl
efxLWHHq8JWceE2ElmCJO
sVF3xl0TR9V8/5pPj+G42
XYAAAA
```

Figure 30: Compression of the condensed varian of GlycoCT. (A) GlycoCT{condensed} 216 characters. (B) GlycoCT{compressed} 202 characters. The reduction of space is greater for bigger sequences with more repetitive elements.

5. Sorting GlycoCT – *canonicalization*

As GlycoCT relies on numbering to identify its entities, a correct, sorted and predictable outcome of a canonicalization algorithm for its graph structures is mandatory. Furthermore it is highly desirable for database applications to have an unique textual describtion of complex oligosaccharides, so the format can serve as a primary key in database applications to perform identity requests.

5.1 Graph traversal algorithm

An unique traversal of the graph is accomplished by a sorting algorithm, for an overview see figure 31.

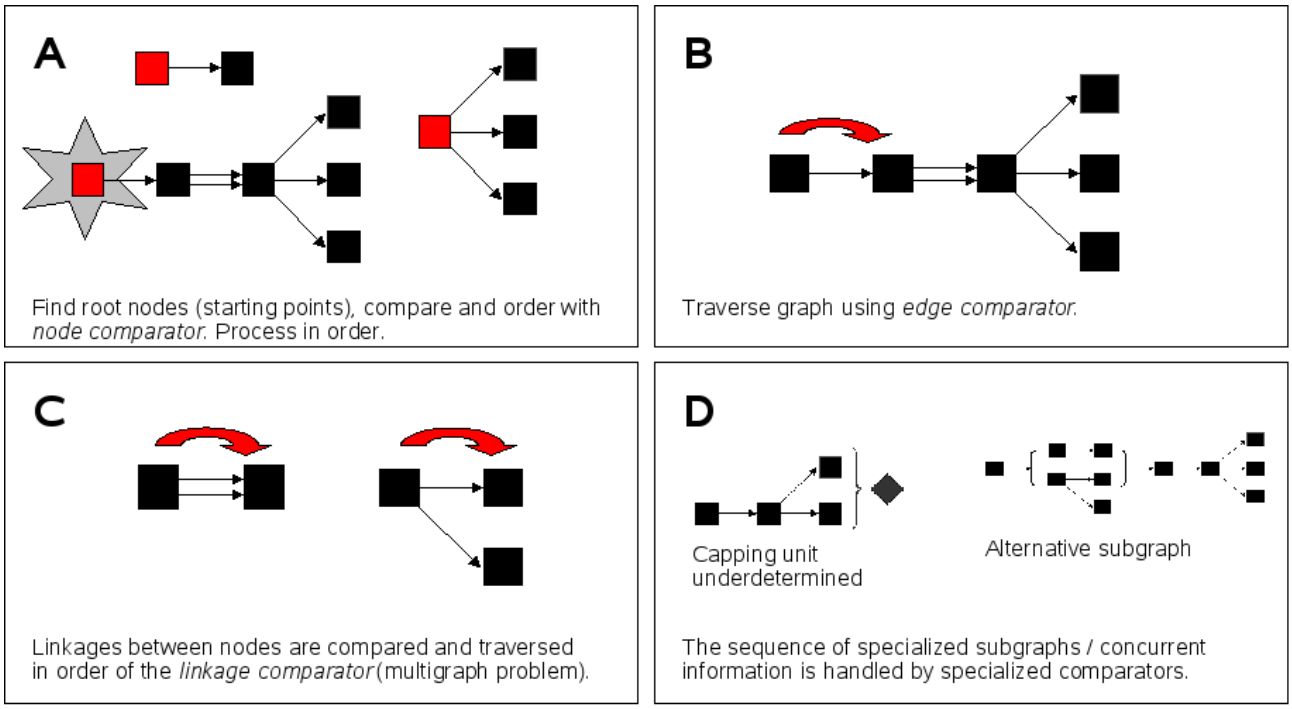


Figure 31: Graph traversal algorithm. (A) The starting point of the traversal is guided by the node comparator (see 5.2). (B) Edges are traversed with the edge comparator, which utilizes mainly the linkage comparator (C).(D) Additional sequence elements are sorted by specialized comparators.

To traverse a sugar graph, the root nodes of a potential forest are traversed in order of the node comparator. Child linkages are traversed in order of the edge comparator. The order of ALT sections is dictated by the alternative unit comparator, whereas concurrent underdetermined subtrees are sorted by the underdetermined subtree comparator.

5.2 Node comparator

Node comparison precedence order. As a fallback an alphanumerical comparison of the resulting GlycoCT – sequences is used.

1. Child residue count
2. Terminal residue count of subgraph

3. Branching point count
4. Alphanumerical comparison of GlycoCT – code with node as starting point

5.3 Edge comparator

Edges of the graph structure may contain multiple linkages (multigraph). Thus the linkage comparator is applied, but first the cardinal number of occurences is taken into account. As a fallback an alphanumerical comparison of the resulting GlycoCT – sequences is used.

1. Cardinal number of linkages
2. Linkage comparator (see 5.4)
3. Alphanumerical comparison of the GlycoCT – sequence of the child nodes

5.4 Linkage Comparator

Linkage comparison precedence order:

1. Numerical parent position comparison
2. Numerical child position comparison
3. Alphanumerical parent linkage type comparison
4. Alphanumerical child linkage type comparison

5.5 Alternative unit comparator (ALT)

The different subgraphs of an alternative unit are compared with the following method:

1. Attach lead-out nodes to subgraph
2. Root node comparison with node comparator

5.6 Underdetermined subtree comparator (UND)

The order of the UND sections, which are concurrent to the main graph structure, is guided by the following order:

1. Root node comparison with node comparator
2. Parent node comparison with node comparator
3. Parent linkage comparison with edge comparator
4. Numerical comparison of the probabilities

6. Examples

Ketoses
The default position for the carbonyl-function in carbohydrates is C1. This is included in the basetype definition. If a keto-function is at another position, the modification identifier „keto“ is used. All occuring keto functions have then to be listed explicitly in the notation. If the keyword „keto“ is found in the modifications, the default C1-carbonyl function is replaced implicitly by a OH-group. Keto functions have an impact on the resulting stereochemistry, as each non-terminal carbonyl-function replaces a stereogenic center. Trivial names for ketoses are deprecated.

IUPAC: D-Fructose, furanose ring, alpha form
CarbBank: a-D-Fruf

GlycoCT:
RES
1b:a-dara-hex-2:5|2:keto

Deoxy sugars
Deoxygenation is indicated with the keyword „d“. Resulting stereoloss is reflected in the basetype name.

IUPAC: 2,6-dideoxy-3-O-methyl-α-D-arabino-hexopyranose
CarbBank: a-2,6-deoxy-D-AraHexp3me

GlycoCT:
RES
1b:a-dara-hex-1:5|2:d|6:d

Acidic sugars
Acidic functions are generally indicated with the letter „a“ in the carbohydrate stem type.

Uronic acids
IUPAC: D-Glucopyranosyluronic acid
CarbBank: b-D-GlcpA

GlycoCT:
RES
1b:b-dglc-hex-1:5|6:a

Aldonic acids
IUPAC: D-Gluconic acid
CarbBank: D-Glc-onic

GlycoCT:
RES
1b:o-dglc-hex-0:0|1:a

Aldaric acids

IUPAC: D-Glucaric acid
CarbBank: D-Glc-aric

GlycoCT: RES

1b:o-dglc-hex-0:0|1:a|6:a

Amino sugars
The mnenomic symbol for amino groups is the „amino“.

IUPAC: 2,6-diamino-2,3,6-trideoxy-α-D-ribo-hexopyranose
CarbBank: a-D-3-deoxy-RibHexp2N6N

GlycoCT:
RES
1b:a-drib-hex-1:5|3:d
2s:amino
3s:amino
LIN
1:1d(2-1)2n
2:1d(6-1)3n

Thio sugars
The mnenomic symbol for thio-functions is the „sh“.

IUPAC: 3-amino-3,4-dideoxy-4-thio-α-D-galactopyranose
CarbBank: a-D-Galp3n4sh

GlycoCT:
RES
1b:a-dgal-hex-1:5
2s:amino
4s:thiol
LIN
1:1d(3-1)2n
2:1d(4-1)3n

Alditols
Reduced sugars are indicated with the keyword „aldi“.

IUPAC: D-Arabinitol
CarbBank: d-Ara-ol

GlycoCT:
RES
1b:o-dara-pen-0:0|1:aldi

Intramolecular anhydrides
Intramolecular anhydrides are written with the special substituent lactone:

IUPAC: 3,6-anhydro-α-D-glucopyranose
CarbBank: 3,6-anhydro-a-D-Glc-p

GlycoCT:
RES
1b:a-dglc-hex-1:5
2s:anhydro
LIN
1:1d(3-6)2o

Unsaturated monosaccharides
„en“ indicates double bonds. The 2 positions identifiers are separated by a delimiting comma.

IUPAC: 2,3-dideoxy-α-D-erythro-hex-2-en-pyranose
CarbBank: a-D-2-en-EryHexp

GlycoCT:
RES
1b:a-dery-hex-1:5|2d|2,3:en|3d

Lactonized carbohydrates
IUPAC: L-xylo-hex-2-ulosono-1,4-lactone (L-Ascorbat, Vitamin C, isomer)
CarbBank: L-Xyl-Hex-2ulo-1,4-lactone-onic

GlycoCT:
RES
1b:o-lxyl-hex-0:0|1:a|2:keto
LIN
1:1d(1-4)1o

Sialic acids

IUPAC: Glycolamidoneuraminic acid (pyranose form) or
5-aminoglycolyl-3,5-dideoxy-D-glycero-α-D-galacto-non-2-ulopyranosonic acid
CarbBank: a-D-NeupGc (D - configuration makes no sense here, but CarbBank did it)

GlycoCT:
RES
1b:a-dgro-dgal-non-2:6|1:a|2:keto|3:d
1s:N-glycolyl
LIN
1:1o(5-1)2

IUPAC: 2-keto-3-deoxy-D-glycero-α-D-galacto-non-ulosonic acid
(KDN, pyranose form)
CarbBank: a-D-Kdnp (D - configuration makes no sense here, but CarbBank did it)

GlycoCT:
RES
1b:a-dgro-dgal-non-2:6|1a|2:keto|3:d

7. Appendix

7.1 XML-schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="sugar">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="residues" type="residueListComplete" minOccurs="1" maxOccurs="1"/>
                <xs:element name="linkages" type="linkageList" minOccurs="1" maxOccurs="1"/>
                <xs:element name="repeat" minOccurs="0" maxOccurs="1" type="repeatUnit"/>
                <xs:element name="underDeterminedSubtrees" minOccurs="0" maxOccurs="1"
type="underDetermined"/>
                <xs:element name="alternative" minOccurs="0" maxOccurs="1" type="alternativeType"/>
            </xs:sequence>
            <xs:attribute name="version" use="required">
                <xs:simpleType>
                    <xs:restriction base="xs:decimal">
                        <xs:enumeration value="1.0"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
        </xs:complexType>
    </xs:element>

    <xs:complexType name="residueListComplete">
        <xs:sequence>
            <xs:choice minOccurs="1" maxOccurs="unbounded">
                <xs:element name="monosaccharide">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="basetype" type="basetypeElement" maxOccurs="unbounded"
minOccurs="0"/>
                            <xs:element name="modification" minOccurs="0" maxOccurs="unbounded"
type="modificationType"/>
                        </xs:sequence>
                        <xs:attribute name="id" use="required" type="xs:positiveInteger"/>
                        <xs:attribute name="anomer" type="anomerType"  use="required"/>
                        <xs:attribute name="superclass" type="residueSuperclass"  use="required"/>
                        <xs:attribute name="ringStart" type="xs:byte"  use="required"/>
                        <xs:attribute name="ringEnd" type="xs:byte"  use="required"/>
                        <xs:attribute name="name" type="xs:string" use="required"/>
                    </xs:complexType>
                </xs:element>
                <xs:element name="substituent">
                    <xs:complexType>
                        <xs:attribute name="id" use="required" type="xs:positiveInteger"/>
                        <xs:attribute name="name" type="xs:string"/>
                    </xs:complexType>
                </xs:element>
                <xs:element name="repeat">
                    <xs:complexType>
                        <xs:attribute name="id" use="required" type="xs:positiveInteger"/>
                        <xs:attribute name="repeatId" type="xs:positiveInteger"/>
                    </xs:complexType>
                </xs:element>
                <xs:element name="alternative">
                    <xs:complexType>
                        <xs:attribute name="id" use="required" type="xs:positiveInteger"/>
                        <xs:attribute name="alternativeId" type="xs:positiveInteger"/>
                    </xs:complexType>
                </xs:element>
            </xs:choice>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="residueListSimple">
        <xs:sequence>
            <xs:choice minOccurs="1" maxOccurs="unbounded">
```

```
                <xs:element name="monosaccharide">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="basetype" type="basetypeElement" maxOccurs="unbounded"
minOccurs="0"/>
                            <xs:element name="modification" minOccurs="0" maxOccurs="unbounded"
type="modificationType"/>
                        </xs:sequence>
                        <xs:attribute name="id" use="required" type="xs:positiveInteger"/>
                        <xs:attribute name="anomer" type="anomerType"  use="required"/>
                        <xs:attribute name="superclass" type="residueSuperclass"  use="required"/>
                        <xs:attribute name="ringStart" type="xs:byte"  use="required"/>
                        <xs:attribute name="ringEnd" type="xs:byte"  use="required"/>
                        <xs:attribute name="name" type="xs:string" use="required"/>
                    </xs:complexType>
                </xs:element>
                <xs:element name="substituent">
                    <xs:complexType>
                        <xs:attribute name="id" use="required" type="xs:positiveInteger"/>
                        <xs:attribute name="name" type="xs:string"/>
                    </xs:complexType>
                </xs:element>
            </xs:choice>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="linkageList">
        <xs:sequence>
            <xs:element name="connection" minOccurs="0" maxOccurs="unbounded">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="linkage" minOccurs="1" maxOccurs="unbounded"
type="linkageElement"/>
                    </xs:sequence>
                    <xs:attribute name="id" type="xs:positiveInteger" use="required"/>
                    <xs:attribute name="parent" type="xs:positiveInteger" use="required"/>
                    <xs:attribute name="child" type="xs:positiveInteger" use="required"/>
                </xs:complexType>
            </xs:element>
        </xs:sequence>
    </xs:complexType>

     <xs:simpleType name="residueType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="b"/>
            <xs:enumeration value="r"/>
            <xs:enumeration value="s"/>
            <xs:enumeration value="a"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="residueSuperclass">
        <xs:restriction base="xs:string">
            <xs:enumeration value="sug"/>
            <xs:enumeration value="tri"/>
            <xs:enumeration value="tet"/>
            <xs:enumeration value="pen"/>
            <xs:enumeration value="hex"/>
            <xs:enumeration value="hep"/>
            <xs:enumeration value="oct"/>
            <xs:enumeration value="non"/>
            <xs:enumeration value="dec"/>
            <xs:enumeration value="s11"/>
            <xs:enumeration value="s12"/>
            <xs:enumeration value="s13"/>
            <xs:enumeration value="s14"/>
            <xs:enumeration value="s15"/>
            <xs:enumeration value="s16"/>
            <xs:enumeration value="s17"/>
            <xs:enumeration value="s18"/>
            <xs:enumeration value="s19"/>
            <xs:enumeration value="s20"/>
        </xs:restriction>
    </xs:simpleType>
```

```
<xs:simpleType name="anomerType">
    <xs:restriction base="xs:string">
        <xs:enumeration value="a"/>
        <xs:enumeration value="b"/>
        <xs:enumeration value="x"/>
        <xs:enumeration value="o"/>
    </xs:restriction>
</xs:simpleType>

<xs:complexType name="basetypeElement">
    <xs:attribute name="id" use="required" type="xs:positiveInteger"/>
    <xs:attribute name="type" use="required">
        <xs:simpleType>
            <xs:restriction base="xs:string">
                <xs:enumeration value="dgro"/>
                <xs:enumeration value="lgro"/>
                <xs:enumeration value="dery"/>
                <xs:enumeration value="lery"/>
                <xs:enumeration value="drib"/>
                <xs:enumeration value="lrib"/>
                <xs:enumeration value="dara"/>
                <xs:enumeration value="lara"/>
                <xs:enumeration value="dall"/>
                <xs:enumeration value="lall"/>
                <xs:enumeration value="dalt"/>
                <xs:enumeration value="lalt"/>
                <xs:enumeration value="dglc"/>
                <xs:enumeration value="lglc"/>
                <xs:enumeration value="dman"/>
                <xs:enumeration value="lman"/>
                <xs:enumeration value="dthr"/>
                <xs:enumeration value="lthr"/>
                <xs:enumeration value="dxyl"/>
                <xs:enumeration value="lxyl"/>
                <xs:enumeration value="dlyx"/>
                <xs:enumeration value="llyx"/>
                <xs:enumeration value="dgul"/>
                <xs:enumeration value="lgul"/>
                <xs:enumeration value="dido"/>
                <xs:enumeration value="lido"/>
                <xs:enumeration value="dgal"/>
                <xs:enumeration value="lgal"/>
                <xs:enumeration value="dtal"/>
                <xs:enumeration value="ltal"/>
                <xs:enumeration value="xgro"/>
                <xs:enumeration value="xthr"/>
                <xs:enumeration value="xery"/>
                <xs:enumeration value="xara"/>
                <xs:enumeration value="xrib"/>
                <xs:enumeration value="xlyx"/>
                <xs:enumeration value="xxyl"/>
                <xs:enumeration value="xall"/>
                <xs:enumeration value="xalt"/>
                <xs:enumeration value="xman"/>
                <xs:enumeration value="xglc"/>
                <xs:enumeration value="xgul"/>
                <xs:enumeration value="xido"/>
                <xs:enumeration value="xtal"/>
                <xs:enumeration value="xgal"/>
            </xs:restriction>
        </xs:simpleType>
    </xs:attribute>
</xs:complexType>

<xs:complexType name="modificationType">
    <xs:attribute name="pos_one" use="required" type="xs:byte"/>
    <xs:attribute name="pos_two" use="optional" type="xs:byte"/>
    <xs:attribute name="type" use="required">
        <xs:simpleType>
            <xs:restriction base="xs:string">
                <xs:enumeration value="a"/>
                <xs:enumeration value="d"/>
                <xs:enumeration value="keto"/>
```

```
                    <xs:enumeration value="aldi"/>
                    <xs:enumeration value="en"/>
                    <xs:enumeration value="enx"/>
                    <xs:enumeration value="sp2"/>
                    <xs:enumeration value="sp"/>
                    <xs:enumeration value="geminal"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
    </xs:complexType>

    <xs:complexType name="linkageElement">
        <xs:sequence>
            <xs:element name="parent">
                <xs:complexType>
                    <xs:attribute name="pos" type="xs:byte"/>
                </xs:complexType>
            </xs:element>
            <xs:element name="child">
                <xs:complexType>
                    <xs:attribute name="pos" type="xs:byte"/>
                </xs:complexType>
            </xs:element>
        </xs:sequence>
        <xs:attribute name="id" use="required" type="xs:positiveInteger"/>
        <xs:attribute name="parentType" use="required" type="linkageType"/>
        <xs:attribute name="childType" use="required" type="linkageType"/>
    </xs:complexType>

    <xs:simpleType name="linkageType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="h"/>
            <xs:enumeration value="d"/>
            <xs:enumeration value="o"/>
            <xs:enumeration value="x"/>
            <xs:enumeration value="n"/>
            <xs:enumeration value="s"/>
            <xs:enumeration value="r"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:complexType name="repeatUnit">
        <xs:sequence>
            <xs:element name="unit" minOccurs="1" maxOccurs="unbounded">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="residues" type="residueListComplete" minOccurs="1"
maxOccurs="1"/>
                        <xs:element name="linkages" type="linkageList" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="internalLinkage" minOccurs="0" maxOccurs="1">
                            <xs:complexType>
                                <xs:sequence>
                                    <xs:element name="linkage" minOccurs="1" maxOccurs="unbounded"
type="linkageElement"/>
                                </xs:sequence>
                                <xs:attribute name="id" type="xs:positiveInteger" use="required"/>
                                <xs:attribute name="parent" type="xs:positiveInteger"
use="required"/>
                                <xs:attribute name="child" type="xs:positiveInteger"
use="required"/>
                            </xs:complexType>
                        </xs:element>
                    </xs:sequence>
                    <xs:attribute name="id" use="required" type="xs:positiveInteger"/>
                    <xs:attribute name="minOccur" use="required" type="xs:int"/>
                    <xs:attribute name="maxOccur" use="required" type="xs:int"/>
                </xs:complexType>
            </xs:element>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="underDetermined">
        <xs:sequence>
            <xs:element name="tree">
```

```
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="residues" type="residueListSimple" minOccurs="1"
maxOccurs="1"/>
                        <xs:element name="linkages" type="linkageList" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="parents" minOccurs="1" maxOccurs="1">
                            <xs:complexType>
                                <xs:sequence>
                                    <xs:element name="parent" minOccurs="0" maxOccurs="unbounded">
                                        <xs:complexType>
                                            <xs:attribute name="res_id" type="xs:positiveInteger"/>
                                        </xs:complexType>
                                    </xs:element>
                                </xs:sequence>
                            </xs:complexType>
                        </xs:element>
                        <xs:element name="connection" minOccurs="0" maxOccurs="1">
                            <xs:complexType>
                                <xs:sequence>
                                    <xs:element name="linkage" minOccurs="1" maxOccurs="unbounded"
type="linkageElement"/>
                                </xs:sequence>
                                <xs:attribute name="id" type="xs:positiveInteger" use="required"/>
                                <xs:attribute name="parent" type="xs:positiveInteger"
use="required"/>
                                <xs:attribute name="child" type="xs:positiveInteger"
use="required"/>
                            </xs:complexType>
                        </xs:element>
                    </xs:sequence>
                    <xs:attribute name="probUp" use="required" type="xs:double"/>
                    <xs:attribute name="probLow" use="required" type="xs:double"/>
                    <xs:attribute name="id" use="required" type="xs:positiveInteger"/>
                </xs:complexType>
            </xs:element>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="alternativeType">
        <xs:sequence>
            <xs:element name="unit" minOccurs="1" maxOccurs="unbounded">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="substructure" minOccurs="2" maxOccurs="unbounded">
                            <xs:complexType>
                                <xs:sequence>
                                    <xs:element name="residues" type="residueListSimple"
minOccurs="1" maxOccurs="1"/>
                                    <xs:element name="linkages" type="linkageList" minOccurs="1"
maxOccurs="1"/>
                                    <xs:element name="lead_in" minOccurs="0" maxOccurs="1">
                                        <xs:complexType>
                                            <xs:attribute name="residue_id" use="required"
type="xs:positiveInteger"/>
                                        </xs:complexType>
                                    </xs:element>
                                    <xs:element name="lead_out" minOccurs="0" maxOccurs="unbounded">
                                        <xs:complexType>
                                            <xs:attribute name="residue_id" use="required"
type="xs:positiveInteger"/>
                                            <xs:attribute name="connected_to" use="required"
type="xs:positiveInteger"/>
                                        </xs:complexType>
                                    </xs:element>
                                </xs:sequence>
                            </xs:complexType>
                        </xs:element>
                    </xs:sequence>
                    <xs:attribute name="id" use="required" type="xs:positiveInteger"/>
                </xs:complexType>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
```

```
</xs:schema>
```