BEILSTEIN JOURNAL OF ORGANIC CHEMISTRY

A consensus-based and readable extension of *Li*near *Co*de for *R*eaction *R*ules (LiCoRR)

Benjamin P. Kellman‡, Yujie Zhang‡, Emma Logomasini, Eric Meinhardt, Karla P. Godinez-Macias, Austin W. T. Chiang, James T. Sorrentino, Chenguang Liang, Bokan Bao, Yusen Zhou, Sachiko Akase, Isami Sogabe, Thukaa Kouka, Elizabeth A. Winzeler, Iain B. H. Wilson, Matthew P. Campbell, Sriram Neelamegham, Frederick J. Krambeck, Kiyoko F. Aoki-Kinoshita and Nathan E. Lewis*

Commentary

Open Access

Address:
See end of main text.

Email:
Nathan E. Lewis* - nlewisres@ucsd.edu

* Corresponding author ‡ Equal contributors

Keywords:
glycoinformatics; linear code; systems glycobiology

Beilstein J. Org. Chem. **2020,** *16,* 2645–2662.
https://doi.org/10.3762/bjoc.16.215

Received: 01 June 2020
Accepted: 17 September 2020
Published: 27 October 2020

This article is part of the thematic issue "GlycoBioinformatics".

Guest Editor: F. Lisacek

Abstract

Systems glycobiology aims to provide models and analysis tools that account for the biosynthesis, regulation, and interactions with glycoconjugates. To facilitate these methods, there is a need for a clear glycan representation accessible to both computers and humans. Linear Code, a linearized and readily parsable glycan structure representation, is such a language. For this reason, Linear Code was adapted to represent reaction rules, but the syntax has drifted from its original description to accommodate new and originally unforeseen challenges. Here, we delineate the consensuses and inconsistencies that have arisen through this adaptation. We recommend options for a consensus-based extension of Linear Code that can be used for reaction rule specification going forward. Through this extension and specification of Linear Code to reaction rules, we aim to minimize inconsistent symbology thereby making glycan database queries easier. With a clear guide for generating reaction rule descriptions, glycan synthesis models will be more interoperable and reproducible thereby moving glycoinformatics closer to compliance with FAIR standards. Here, we present Linear Code for Reaction Rules (LiCoRR), version 1.0, an unambiguous representation for describing glycosylation reactions in both literature and code.

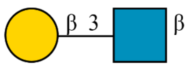
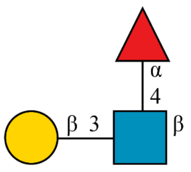
Introduction

Glycans are predominantly synthesized through the serial addition of monosaccharides to form large polysaccharides. To build computational models of glycan synthesis, the biochemical reactions involved must be defined and described mathematically in a form that can be interpreted by computers [1-3]. Several groups have created such models using a variety of strategies, including mechanistic and nonlinear [4-12], linear probabilistic [13,14], machine learning [15], formal-grammar [16], and substructural [17]. Unfortunately, most of these approaches use slightly different expressions of the building blocks, the reaction rules, therefore, model comparison is more challenging than it needs to be, with certain inconsistencies remaining to be resolved.

In the past few decades, substantial efforts made in the construction of these models of glycan synthesis were mostly focused on defining reaction rules that benefit from an unambiguous representation with human readability. For example, graphical denotation is one of the most human-understandable representations to describe reaction rules [18-20]. While graphical representations are intuitive and extremely accessible to a human reader, they are not computationally accessible due to ambiguities in their representations. There are already efforts to create computationally transmissible rule sets in XML-type representations like BioPAX [21], CellML [22], and SBML [7,8] which are readily interoperable and reusable. However, the XML-type model representations are not designed to be human-readable or included in the main text of a manuscript confining many design details to the supplement of a publication. As systems glycobiology develops, there is a need to develop a standard nomenclature for unambiguous and readable reaction rules to facilitate development, exchange, extension, and validation of glycosylation models and analysis tools.

Here we bring explicit attention to the concerns we raise above, we provide a focused, text-based representation of reaction rules that have been introduced for the purpose of formalizing these communications. GlycoCT [23] and WURCS [24,25] are two popular glycan nomenclatures in use today. GlycoCT was designed to maximize the descriptive specificity of the experimentally derived glycan structures data. WURCS, on the other hand, focuses on the uniqueness of a linear representation which promises efficient lookup in database queries. Both GlycoCT and WURCS produce unambiguous representations and are thereby invaluable for many applications, ranging from systems biology analyses [17] to an international glycan structure repository [26-29]. GlycoCT and WURCS provide a high degree of unambiguous detail; however, they are limited in their human-readability. The glycan extension to IUPAC, on the other hand, is more human-readable [30]. It specifies the linkage and branch information in an intuitive and linear manner. In the hopes of mitigating the inconsistent application of IUPAC and inconvenient illustrations, Linear Code described a simplified version of IUPAC nomenclature [31]. Specifically, Linear Code is a syntax for representing glycoconjugates and their associated molecules in a simple linear fashion. While keeping the linkage and branch information, Linear Code removes the hyphens between monosaccharides and abbreviates the glycan symbols, thereby simplifying the representation without limiting flexibility. Given its readability and parsability, Linear Code has become a popular choice for representing reaction rules in computational models of glycan synthesis (Table 1). However,

Table 1: The reaction rule Ab3GNb → Ab3(Fa4)GNb represented in Symbol Nomenclature for Glycans [18], Linear Code, IUPAC, GlycoCT, and WURCS separately. Linear Code provides the most straightforward and succinct representation.

	Reactant	Product
Structure plot (with link info)		
Linear Code	Ab3GNb	Ab3(Fa4)GNb
IUPAC-extended	β-D-Gal*p*-(1-3)-β-D-Glc*p*2NAc	β-D-Gal*p*-(1-3)-[α-L-Fuc*p*-(1-4)]β-D-Glc*p*2NAc
IUPAC-condensed	Gal(β1-3)GlcNAc(β1-	Gal(β1-3)[Fuc(α1-4)]GlcNAc(β1-
glycoCT	RES 1b:b-dglc-HEX-1:5 2s:n-acetyl 3b:b-dgal-HEX-1:5 LIN 1:1d(2+1)2n 2:1o(3+1)3d	RES 1b:b-dglc-HEX-1:5 2s:n-acetyl 3b:b-dgal-HEX-1:5 4b:a-lgal-HEX-1:5|6:d LIN 1:1d(2+1)2n 2:1o(3+1)3d 3:1o(4+1)4d
WURCS	WURCS=2.0/2,2,1/[a2122h-1b_1-5_2*NCC/3=O] [a2112h-1b_1-5]/1-2/a3-b1	WURCS=2.0/3,3,2/[a2122h-1b_1-5_2*NCC/3=O] [a2112h-1b_1-5][a1221m-1a_1-5]/1-2-3/a4-c1_a3-b1

with the rise of Linear Code adaptations to represent reaction rules, we have seen increasing diversity in the syntax, including branch constraints, duplicate monosaccharides omission, logical operators, etc.

Here we critically review reaction rule nomenclature. In doing so, we seek to promote the development of a standardized and unambiguous, readable, and computable reaction rule representation. First, we examine the original usage of Linear Code for reaction rule representation by discussing six major categories of syntax rules. Second, we discuss the various adaptations that have been introduced in the current usage of Linear Code to represent reaction rules. Third, we further discuss the apparent nomenclature ambiguity emerging in the adaptation of Linear Code to systems glycobiology. Finally, we demonstrate the depth of the nomenclature crisis through the minimal overlap in presumably similar networks. While many solutions to this nomenclature might be offered, we focus on six major recommendations to provide a unified representation of reaction rules that are likely to have a broad impact on minimizing change to the current adaptations.

Common lore at universities describes architects who, rather than "prescribe" ideal paths for students through the mall, waited to see where students would walk. They built their paths over the trampled grass of the "descriptive" paths chosen by the students. Similarly, we intend to extend the thoughtful "prescription" of Linear Code to "descriptive" extensions that will comfortably accommodate those currently working in systems glycobiology. We also provide some key definitions for ease of reading (Figure 1, Table 2).

Syntax Rules of the Original Linear Code

Linear Code rules can be separated into six categories of syntax rules (Table 3): Stereospecificity and ring structure rules (SRS), modification rules (MR), branch rules (BR), repetition rules (RR), glycoconjugate rules (GR), and uncertainty rules (UR). The saccharide unit (SU) refers to a structure with four elements: anomericity, position number, modifications, and monosaccharide (MS).

Stereospecificity and ring structure rules are set to differentiate the stereoisomers or distinct ring structures. A change from primary to secondary stereospecificity is denoted by " ' ", while a change to secondary ring structure is denoted " ^ ". A change to both secondary ring and stereospecificity is denoted " ~ ". For example, " G " represents glucopyranose, the pyranose con-

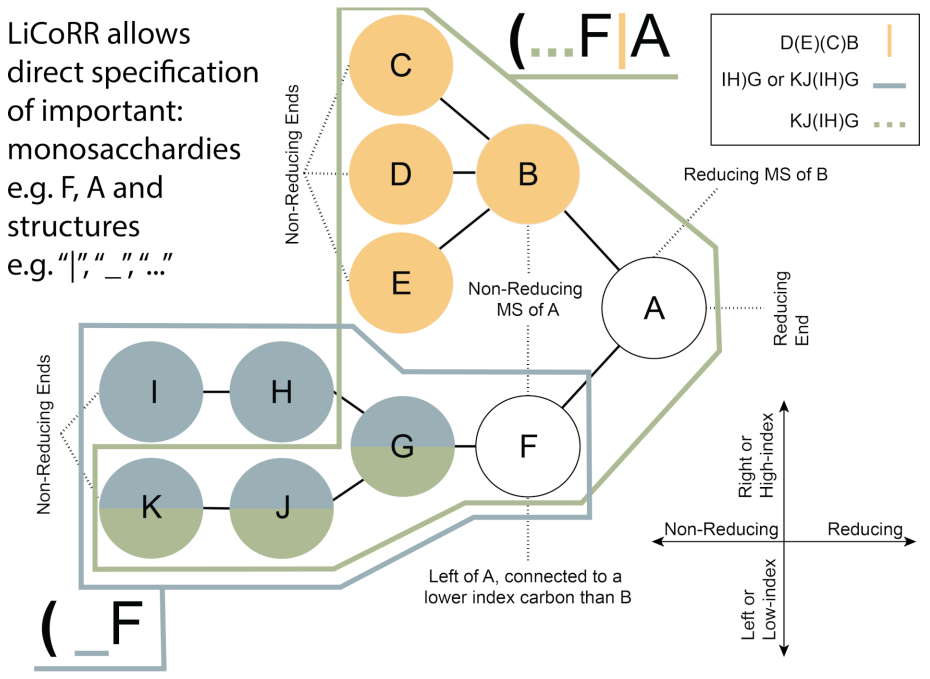


Figure 1: Common terminology and anatomy of a theoretical glycan, (KJ(IH)GF(D(E)(C)B)A. In this figure, we demonstrate some key terminology as well as the three primary uncertainty operators: branch (orange), continuation (blue), and ligand (green). The structures matching these terms are shown in matching colors, those matching both the continuation and ligand are shown in purple. A ligand can typically be removed with one cut. A continuation is a connection from a node to a root that can "continue" or "bypasses" other branch points. The paths from I to G or K to G represent one continuation; to represent both paths, a continuation is necessary because traversing from I to G requires the syntactic "bypass" of the KJ branch.

Table 2: Glossary of essential terms.

Term	Definition
saccharide unit (SU)	composed of a monosaccharide name, modifications (if any), anomericity (α or β configurations of the glycosidic bond), and the position it is bonded to a given SU.
monosaccharide (MS)	a sugar monomer.
lowest-carbon-index chain	the lowest carbon index branch corresponding to the non-reducing sugar connected to the lowest reducing-end carbon.
branch	any right branch, pictorially “right” of the reducing MS (Figure 1), where a non-reducing sugar is not connected to the lowest reducing-end carbon.
reducing and non-reducing ends	these are the MSs that appear “first” (closest to the glycoconjugate or first added in the synthesis) and “last” (leaves or terminal MS, those farthest from the “first” MS within a branch and have no linkage to a non-reducing MS). Typically, there is one reducing end and there are often multiple non-reducing ends.
reducing MS	closer to the first MS or the “reducing-end”.
non-reducing MS	farther from the first MS and closer to a non-reducing end.

Table 3: Original Linear Code rules (Banin et al. [31]).[a]

	Rule description	Example
saccharide unit (SU)	1. see one-letter MS names in Table 4.	
	2. the anomer, where an α conformation is denoted as “a,” and β as “b,” follows the one-letter MS name.	Ga, Gb
	3. the carbon number by which the SU is attached follows the anomer.	Ga3, Gb2
	4. modifications. see modification rules for details, which follow after the carbon number.	see modification rules examples.
open form (OF)	1. open form notation. If a carbon is in its open-chain form, an “o” is attached to the end.	AbGo, AbG[P]o
stereospecificity and ring structures (SRS)	1. the less common stereoisomer (D or L) of an MS is indicated with apostrophes (‘).	D-Glc*p*: G L-Glc*p*: G’
	2. MSs with uncommon ring structures (e.g., furanose, pyranose) are indicated with a caret (^).	D-Glc*p*: G D-Glc*f*: G^
	3. MS that differ in both common stereospecificity and ring structure are indicated with a tilde (~).	D-Glc*p*: G L-Glc*f*: G~
modification rules (MR)	1. the modifications are represented by adding square brackets that include the connecting position of the modification to the SU, followed by the modification symbol (Table 5) in the form: [<number><symbol>] Exceptions include certain monosaccharides with common modifications (D-GalpNAc is AN instead of A[2N]). Anomericity (α or β) is expressed immediately after the modification.	β-D-Gal*p*(2P)-(1-3)-β-D-Glc*p*: A[2P]b3Gb
branch rules (BR)	1. when the non-reducing MSs are identical, the MS linked to the higher index carbon will branch (appear first in the written representation when read right to left, reducing to non-reducing end).	GNb2Ma3(NNa3Ab3GNb2Ma6)Mb4GNb
	2. when the non-reducing MSs are different, the less frequent non-reducing MS will branch (MS frequency Table 4).	Ab3ANb4(NNa3)Ab4Gb
repetition rules (RR)	1. repeating units are expressed inside parentheses, with an ‘n’ representing the number of repeats.	cellulose, which is a polymer of D-glucose residues joined by β-1,4 linkages are represented as {nGb4}
	2. if not the non-reducing end, the head of a repeated motif is expressed two dashes “ - - ”	{nGa6Ga4(-Ab3-)Ub2Ha3Ha3Ha3}
	3. if not the reducing end, the tail of a cyclic motif is expressed using the letter “c”.	nGa6Ga4(-Ab3-)Ub2Ha3Hc̲a3Ha3

Table 3: Original Linear Code rules (Banin et al. [31]).[a] (continued)

glycoconjugate rules (GR)	1. amino acid sequences are written after ‘;’. Lipid moieties are written after ‘:’. Other glycosides are written after ‘#’.	Ga;NY-S-C. Gb:C GNb3Ab#4-Trifluoroacetamidophenol
uncertainty rules (UR)	1. α or β linkage unknown, or connection position unknown: ?	AN?3G
	2. both linkage and connection position unknown: ??	AN??G
	3. an entire SU unknown: * * could match any whole SU.	ANb3*A
	4. when two possibilities are given for the identity of an SU element, use “/”	ANb3/4
	5. when two options are given for the identity of a complete SU, use “//”	Ab4//Ga2Aa3 represents Ab4Aa3 or Ga2Aa3
	6. for glycan fragments, use an index number + ‘%’ as a variable for the fragment, and a ‘|’ to separate the fragment from the core.	NNa6=1%|1%Ab4GNb2Ma3(1%Ab4GNb2Ma6)Mb4Gb denotes that Ab4GNb2Ma3(Ab4GNb2Ma6)Mb4Gb is the core, and that the linkage of the fragment NNa6 to the core is uncertain. % means uncertain, 1 is the index referring to the uncertain MS.

[a]“(#)” - Rules deprecated in LiCoRR.

formation of glucose, with D stereospecificity. Glucopyranose with L stereospecificity is written as “ G’ ” (SRS1). Glucofuranose with D stereospecificity is written as “ G^ ” (SRS2), and glucofuranose with L specificity is written as “ G~ ” (SRS3). Similarly, galactofuranose, a common fungal monosaccharide, would be written “A^”

Open form rule indicates that if the MS at the reducing end is open – a linear rather than cyclic MS, then the final character to the right of the string should be "o". For example, lactose, galactose β-linked to glucose would be written as AbG if the reducing end glucose is closed and AbGo if the glucose is open; the open "o" takes the place of the linkage in this context. If the glucose is phosphorylated, this structure would be written AbG[P]o.

Modification rules specify a modification of a MS at certain positions (MR1). MS + “ [” + modification + “] ” is used to denote the modification. For example, “G[2S]” describes sulfation on the second carbon of a D-glucopyranose. The anomericity is expressed to the right of the modification (i.e., “G[2S]a”). Multiple modifications to the same MS are ordered based on the position number inside the same brackets; ascending order from left to right. For some common modifications like *N*-acetylgalactosamine, instead of “A[2N],” Linear Code uses “AN” directly. Table 4 includes syntaxes of MS in Linear Code and common modified MSs. Common modification names can be found in Table 5. Given multiple modifications, carbon numbers are written in ascending alphanumeric order. Therefore, dideoxy galactose, or abequose, is written “A[2,6D]” while *N*-aceytlfucosamine could be written “A[6D,2N]”.

Branch rules specify which non-reducing saccharide unit (SU) should be in the branch and which SU should continue the lowest-carbon-index chain; branching is determined by the identity of the first MS in a chain. When the non-reducing MSs are identical, the MS and its substituent chain, linked to the higher carbon of the reducing MS, will branch while the MS and substituent chain, linked to the lower carbon position of the same reducing MS, remains in the lowest-carbon-index chain (BR1). Otherwise, if the non-reducing MSs are different, the chain with a less frequent non-reducing MS (lower rank in Table 4) is considered the branch (BR2). The MS frequency is specified in Table 4, decreasing from top to bottom. When there are more than two non-reducing MSs linked to the same reducing MS, they are ranked, first by frequency, then by linkage index. The highest frequency MS is ranked higher, further to the left when the expression is written. Any MSs with equal rank after the frequency rank – those that are the same MS – are ranked by their linkage index, the lowest linkage indexes are ranked higher. A higher rank means these MSs, and their associated chains, will remain on the lowest-carbon-index chain, while the lower rank MSs will branch.

Repetition rules specify the contraction syntax for succinctly describing repeating MS units. The repetition structure is denoted by curly brackets, with a prefix of repetition times inside the brackets. For example, cellulose, which is a polymer of D-glucose residues joined by β-1,4 linkages, is represented as

Table 4: Common monosaccharides and their Linear Codes (adapted from [31]). We have added NG as it has become a clearly important monosaccharide excluded from the original list. Full monosaccharide descriptions are recorded in IUPAC [18]; all terms can be found at https://www.qmul.ac.uk/sbcs/iupac/2carb/38.html.

Monosaccharides[a]	Linear Code	IUPAC
D-glucose	G	Glc
D-galactose	A	Gal
N-acetylglucosamine	GN	GlcNAc
N-acetylgalactosamine	AN	GalNAc
D-mannose	M	Man
N-acetylneuraminic acid	NN	Neu5Ac
**N*-glycolylneuraminic acid[b]	NG	Neu5Gc
neuraminic acid	N	Neu
2-keto-3-deoxynononic acid	K	KDN[c]
3-deoxy-D-manno-2 octulopyranosylonic acid	W	Kdo
D-galacturonic acid	L	GalA
L-iduronic acid	I	D-IdoA
L-rhamnose	H	Rha
L-fucose	F	Fuc
D-xylose	X	Xyl
D-ribose	B	Rib
L-arabinofuranose	R	Ara*f*
D-glucuronic acid	U	GlcA
D-allose	O	All
D-apiose	P	D-Api
D-fructofuranose	E	Fru*f*
*ascarylose[b]	C	Asc
*ribitol[b]	T	Rib-ol (Rbo)

[a]All the monosaccharides are in their pyranose form unless otherwise noted. [b]Asterisk ("*") represents an update from the original table. [c]KDN: 3-deoxy-D-glycero-D-galacto-nonulosonic acid. Kdn: 3-deoxy-D-glycero-D-galacto-nonulosonic acid.

Table 5: Common modifications and their Linear Code (from [31]).

Modification type	Linear Code	IUPAC
deacetylated *N*-acetyl	Q	N
phosphoethanolamine	PE	Pe
inositol	IN	In
methyl	ME	Me
N-acetyl	N	NAc
O-acetyl	T	Ac
phosphate	P	P
phosphocholine	PC	Pc
pyruvate	PYR	Pyr
sulfate	S	S
sulfide	SH	Sh
aminoethylphosphonate	EP	Ep
*deoxy[a]	D	d
*carboxylic acid[a]	CA	-oic
*amine[a]	A	-amine
*amide[a]	AO	-amide
*ketone[a]	K	-one

[a]Asterisk ("*") represents an update from the original table.

"{nGb4}" (RR1). If a ring structure is repeated and the repeating unit is not connected "head to tail," the MS where the repeating units are connected is marked between 2 dashes " - - " (RR2). An example is {nGa6Ga4(-Ab3-)Ub2Ha3Ha3Ha3}. Additionally, Banin et al. specify that a cyclic motif, a form of repetition, is expressed using the letter "c" [31]. While specification was limited in the original publication, we interpret "c" as denoting the "tail." (-X-) denotes the head if it is not the left end and "c" denotes the tail if it is not the right end of the string. For example, in the molecule nGa6Ga4(-Ab3-)Ub2Ha3Ha3Ha3, Ab3 connects to the reducing end, Ha3. But if Ab3 was connected to the second Ha3 from right instead, we can specify the point of the cycle using a "c," nGa6Ga4(-Ab3-)Ub2Ha3Hca3Ha3.

Glycoconjugate rules describe when a reducing end of a SU is connected to non-carbohydrate moieties, Glycoconjugate rules regulate that amino acid sequences are written after " ; ", lipid moieties are written after " : ", and other glycosides are written after " # " (GR1). For example, a glucose β-linked to a Ceramide is written as "Gb:C."

Uncertainty rules describe syntax for when certain features of the SU are unknown or have more than one possibility. If the anomericity of certain bonds is unknown, Linear Code uses " ? " (i.e., AN?3G) (UR1). If both linkage anomericity and position are unknown, Linear Code uses " ?? " (i.e., AN??G) (UR2). If an entire SU is unknown, " * " can be used instead. ANb3*A represents a three SU glycan, where the second SU is unknown (UR3). When two monosaccharides are possible for a given SU, Linear Code uses the forward slash to separate them. When SU ambiguity refers to anomericity, position number, modifications, or MS, a single " / " is used (i.e., ANb3/4) (UR4). Given two complete possible SUs, Linear Code uses " // " to separate them (i.e., Ab4//Ga2Aa3 represents Ab4Aa3 or Ga2Aa3) (UR5). When analyzing fragmented glycans, an "< index number>%" is used to store fragmented structures as a variable. For example, NNa6=1%|1%Ab4GNb2Ma3(1%Ab4GNb2Ma6)Mb4Gb is a glycan containing a terminal α-2,6-linked sialic acid (NNa6) whose linkage position is unknown. Here, the " | " is used to separate the fragment(s) and core structure components (UR6).

In the interest of demonstrating the reach of single letter LC monosaccharides (Table 4), we provide a monosaccharide network suggesting demonstrating non-trivial functional-group (Table 5) relations between monosaccharides (Figure 2). We used RDKit, an open-source cheminformatics toolkit, to identify chiral centers and further determine stereochemical equivalence classes. Monosaccharides were clustered with an 80% stereo-similarity threshold (Figure 2A), and the maximum common substructure (MCS) of each cluster was obtained (Figure 2B). These MCS equivalence classes were used to group monosaccharides explicitly listed in Table 4 and connect them through addition or subtraction of functional groups in Table 5 (Figure 2C) to every major monosaccharide listed by SNFG (Figure 2D). Figure 2D shows some of these non-trivial paths (e.g., beyond GlcNac; G → GN or G[2N]) from Table 4 monosaccharides, to all listed SFNG monosaccharides via modifications from Table 5. We further provide a full network (Table S6, Supporting Information File 1) to facilitate the discovery of any monosaccharide–monosaccharide relation. For example, the fucose-galactose relation can be found in row 1479 of Table S6 (Supporting Information File 1). They differ by one hydroxy group therefore fucose could be represented as “A[6D]”. Similarly, abequose, a dideoxy galactose, could be represented as “A[2,6D]” or “F[3D]”. Through simple lookup in Table S6 of Supporting Information File 1, many noncanonical monosaccharides can be described thus mitigating the limitations of the single-letter monosaccharide representation.

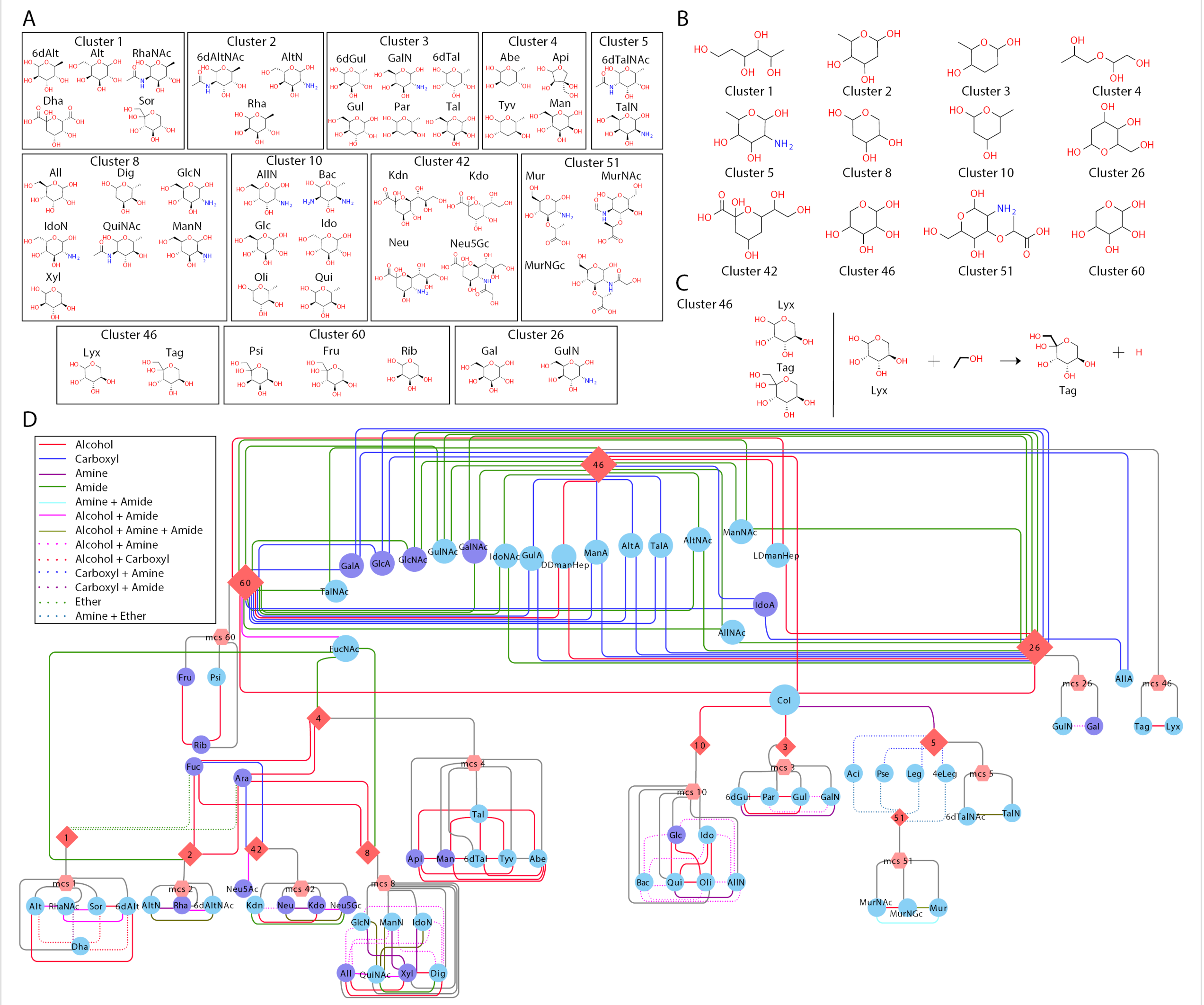


Figure 2: Monosaccharide reachability analysis. (A) Clusters contain monosaccharides with highly similar stereochemistry (>80%). (B) The maximum common substructure (MCS) associated with each cluster. (C) An example to illustrate the modifications needed to reach one monosaccharide to another, as identified by the complete monosaccharide reachability network (Table S6, Supporting Information File 1). (D) The monosaccharides reachability network, showing only connectivity for the least number of modifications needed, differentiated by color as stated in the legend, between monosaccharides (circle) and clusters (diamond). Additionally, the node size denotes the number of different possible paths taken for them to be reached. Please note that each edge is not a predicted or proposed feasible reaction. Edges denote functional groups that can be added or removed from one monosaccharide to represent another.

Current Usage of Linear Code to Represent Reaction Rules

Linear Code was first used to represent reaction rules in 2009. A reaction network, specifying glycans with condensed IUPAC and Linear Code, was trained on mass spectrometry abundance to learn biosynthetic enzyme activities [10]. Their reaction rules table contained four features: enzyme, reactant, product, and constraint. For their implementation, not all original Linear Code rules are adopted. Krambeck et al. [10] maintained the linkage information (Table 3: SU2), one-letter MS abbreviation (Table 4), and branch rules (Table 3: BR), which are the necessary conditions to denote a glycan with branches [10]. On the other hand, symbols " ~ ", " * ", " | " were defined with new meanings, though they already had their meanings in the original Linear Code rules (Table 3: SRS3, UR3, UR6, respectively). Instead, Krambeck et al. introduced several new symbols to convey logical relationships (" & ", " ~ ", "or") and structural ambiguity (" ... ", " _ ", " | ", " * "), all of which were used to specify constraints. For example, a constraint "Ma6 & Ma3" means the reaction will happen only if both Ma6 and Ma3 appear in the glycan; as an N-glycan, these are the terminal mannoses capping the chitobiose core. The "Ma6 or Ma3" constraint promotes the reaction if either Ma6 or Ma3 exists. "~Ma6" means the reaction will not happen if Ma6 is present in the glycan. The structure denotations are indicators of certain parts of the glycan. The entry " ... " can be replaced with either nothing or any polysaccharide with matched parenthesis. The entry " _ ", in Krambeck et al., can be replaced with either nothing or any polysaccharide where each left parenthesis is matched to a right parenthesis but where right parentheses are not necessarily matched. Entry " | " represents a possible branch. We expand on the distinctions between " … ", " _ " and " | " in a later section "Substring uncertainty operators" (Table 4). The asterisk " * " stands for the reaction site, which is the position where the new MS will be added or an MS is removed. Krambeck et al. also uses " # " to describe constraints around the number of MS that may appear in a glycan. For example, the constraint "#A = 0" means the reaction will happen only if there is no galactose. The Krambeck et al. adaptation is the most common adaptation of Linear Code to represent reaction rules [7,13,15,32].

Based on the Linear Code reaction rules framework Krambeck et al. created, later researchers introduced new attributions that specify and simplify the description of reactions. Bennun et al. and Spahn et al. include the amino acid at the end of the Linear Code attached by a semicolon " ; ". This suffix is exactly the syntax from the original Linear Code rules (Table 3: GR1). The reaction rules table generated by Spahn et al. also provided localization information, which is either *cis*, *trans*, or *medial* to denote the Golgi compartment where the reactions happen [13,14]. The subcellular localization of a reaction, in the endoplasmic reticulum, Golgi, cytoplasm (bacteria and archaea), or lysosome (degradation, Man-6-P dephosphorylation and lysosomal glycoprotein biosynthesis [33,34] or paucimannose recycling [35]), are important constraints on glycosylation [36], therefore, the addition of this information to the Linear Code reaction rules provides insights into the glycosylation types.

Some models of glycan synthesis generated reaction rule tables with an additional column Enzyme Commission number (EC number) [7,16,37]. The EC number system is a numerical classification scheme for enzyme-catalyzed reactions that provides an unambiguous accession to a cataloged reaction [38]. The inclusion of an EC number in the reaction rules table, therefore, promotes the clarity, interoperability, and reproducibility of the generated reaction model.

A common syntax used by most studies is the leftmost " (" to represent the terminal, non-reducing end of the glycan chain. It specifies whether the leftmost MS is the terminal MS both visually and computationally. For example, the reaction rule (GN $\rightarrow$ (Ab3GN applies to all reactions which add one galactose to a terminal *N*-acetylglucosamine. On the other hand, the reaction rule GN $\rightarrow$ Ab3GN applies to all reactions which add a galactose to an *N*-acetylglucosamine, but not necessarily the terminal one. The leftmost " (", therefore, can easily vary the glycan substrate substantially.

Though Linear Code was developed with parsability in mind, some have found it useful to make a specific computational implementation of the reaction rules to accommodate the syntactic constraint of programming languages. A human milk oligosaccharide metaglycome was constructed using a combination of linear code, glycan structures represented in XML and XPath queries [39]. Separately, Akune et al. generated a theoretical *N*-glycan database called UniCorn, based upon a Perl implementation of reactions on glycans represented in Linear Code [37]. Though Linear Code is computer-parsable, there is still substantial work necessary to implement that parsing because there is no standard representation for handling the wide variety of reactions possible, nor open-source software available to implement the parsing of such rules.

Representing reaction rules in Linear Code is not easy because of a few ambiguous cases not completely described in the initial Linear Code paper. Subsequent studies, therefore, have developed their own ways to idealize reaction rule implementations based on Linear Code. Using the framework Krambeck et al. built [10], new information like Golgi localization and EC numbers are added to specify and simplify the reaction rules.

Original Prescriptions for Substring Uncertainty Operators

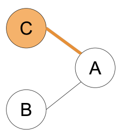
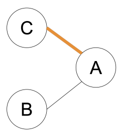
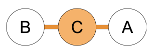
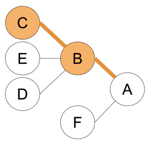
In its original conception [10], the adaptation of Linear Code to represent reaction rules aimed to describe how glycosylation enzymes change the structure of glycans in terms of how the Linear Code character string descriptions of the glycans are changed (Figure 1). In the simplest case, we can specify a substring of the substrate code to be replaced by a new substring to form the product code. In addition, there can be constraint and adjustment substrings whose presence or absence within the substrate string either restricts which glycans can be substrates of a particular enzyme or modifies the reaction rate parameters. Uncertainty operators have been developed to facilitate searching substrings for specific structural features of a glycan implied by the substrings.

The substring specifications for the substrate, product and adjustments can include any combination of characters included in the glycan codes in addition to uncertainty operators inserted within the directly specified characters. Each uncertainty operator is represented by one or more characters, such as "…" or "_" (Table 6). To perform substring matching of a glycan to a substring with uncertainty operators, we first identify the characters of the specified string immediately before and after the uncertainty operator. If found, we then test the substring of the glycan string between these two matched character strings and check for the appropriate uncertainty operator properties. In parsing the glycan code, an initial left parenthesis is always added to the complete glycan code so that the terminal end of every branch of the glycan is always a left parenthesis. Below, in defining the properties of substrings corresponding to an uncertainty operator, we use the symbol X to represent some monosaccharide with its connection, such as Ma3, GNb4, etc.

There are three types of uncertainty operators. The ligand " ... ", the continuation " _ ", and the branch " | ". Each has a specific syntactic match, but intuitively, the ligand is a chain that can contain branches, the continuation is a chain that can include branches terminated outside of the continuation, and the branch is either a complete branch or nothing. Functionally, " ... " indicates the "leftward" extension along the lowest-carbon-index chain, " | " indicates the "rightward" extension along the highest-carbon-index chain, and " _ " indicates an extension along either the left or right chain (Figure 1).

More specifically, (1) the ligand uncertainty operator indicates a chain of MSs that can include attached branches completely contained in the substring, (2) the continuation uncertainty operator indicates a chain of MSs that can include attached branches that may not be wholly contained in the substring, and

Table 6: The difference between " _ ", " … " and " | " with illustrations. These symbols were proposed by Krambeck et al. [10]. The initial names are ligand (" ... "), continuation (" _ "), and possible branch (" | "). Each uncertainty operator in the last four example columns can be replaced by the substring in red to achieve the behavior described in the column header. For a more comprehensive look at the usage of these uncertainty operators, see Supporting Information File 1, Table S1 for a manual collection of matches, and Table S4 (Supporting Information File 1) for an automated collection of matches.[a]

Symbols	Syntax	Meaning	Add a whole new branch	Initiating branch	Extending lowest-carbon-index chain	Initiating nested branch
			B*(C)*A	**B(C)A**	**B*C*A**	**E(D*(C)B*)A**
						
_	any string where every '(' has a matching ')'. Includes the empty string.	chain bypassing a branch to reach a reducing MS; A continuation cannot necessarily be removed by splitting one linkage (can contain branches)	B_A	B(C_A	B_A	E(D_A
...	any string with all matching parentheses. Includes the empty string.	chain to a reducing MS; A ligand can typically be removed by splitting one linkage (can contain branches)	B...A		B...A	
+ (formerly "|")	')' or '(...)' or ')(...)'. Or an empty string.	possible branch point.	B+A	B(C+A		

[a]A, B, C, D, E are abstract monosaccharides.

(3) the possible branch uncertainty operator indicates where a branch may be included in the substring. Due to the nuances of representing a glycan linearly, these are not complete definitions.

Ligand "…" – A ligand is a fragment of a larger molecule connected to the rest of the molecule at one point. Glycans are themselves ligands, as they are pieces of larger molecules. A substring is a valid "ligand" if each parenthesis in that substring is uniquely and appropriately matched; each left parenthesis must be followed by a corresponding right parenthesis and each right parenthesis must be preceded by a corresponding left parenthesis: ")(" are not matched parentheses. Any substring with all left and right parentheses matched, including an empty string, is considered a ligand. If we select a substring of the code representing a glycan it may or may not represent a ligand. For example, XXX)X, XX(XX, X)(XX are not valid ligands, while XX(XXX)XX is valid.

Functionally, ligands can serve as connectors between the left and right portions of a glycan a user would like to specify. A ligand is simply a chain of monosaccharides which may contain nested branches; the nested branches must also be ligands. However, there can be many chains or paths through a ligand, starting from one of the terminal monosaccharides and culminating at the root end; there are many ligands within most ligands. Any of these ligands can serve as a connector from the root (reducing) end of one ligand to a terminal (non-reducing) end of another. The key property of a ligand substring is that all the included branches of the ligand are completely contained in the substring.

Continuation "_" – As we parse from left to right through a substring, we may find left parenthesis (entering into a branch) and right parenthesis (exiting a branch). A ligand, with matched parentheses, indicates an equal number of branch initiations and completions. On the other hand, a substring with an unmatched right parenthesis, for example, XXX)XX or X)(XX)XX, indicates a net termination of branching; each right parenthesis indicates moving out of a branch towards a root. As long as all the left parentheses encountered are followed by right parentheses, we are following a path along a connected chain of the glycan structure. A substring where every left parenthesis can be matched with a following right parenthesis, but not necessarily vice versa, is a "continuation." Again, we include the empty string in this class of substrings. Note that any ligand is also a continuation.

The continuation uncertainty operator can be very useful in formulating rules that apply to specific monosaccharides connected by a chain of monosaccharides to a particular reducing monosaccharide of the glycan structure. For example, the iGnT enzyme adds a Gnb3 group to a terminal galactose group and has a preference for the two branches that are connected to the Ma6 of the root Ma3(Ma6)Mb4 structure. This leads to an adjustment rule based on the string Ma3|(*_Ma6)Mb4. Here the "|" uncertainty operator is used to allow for the possible presence of a bisecting GlcNAc on the root mannose: Ma3(GNb4)(…Ma6)Mb4. The "*" indicates the site of the enzyme action.

Possible branch "|" – As discussed, parsing a linear glycan from left to right, we can encounter matched parentheses indicative of a ligand or unmatched right parentheses indicative of a closing branch. We can leverage the branch closer offered by these symbols to mandate a possible branch. The definition of the "possible branch" is one of either: ") ", " (...) ", ")(…) " or an empty string. This uncertainty operator can be replaced with either a branch, the start of a branch or nothing. It allows the same specification string to work whether an additional branch is present at the position of the uncertainty operator or not, as in the above example.

Divergence in Current Implementations of Reaction Rules from Original Linear Code

Linear Code is a useful notation to succinctly describe glycan structures. It is thus useful to represent substrates and products. However, constraints on the glycan acceptor class where a new monosaccharide is added, was beyond the scope of the original Linear Code rules. Therefore, different adaptations are introduced throughout the literature.

We have identified four symbols that are prescribed with different meanings than they were originally assigned in the Linear Code rules:

Ambiguous symbol 1 – Originally, " ~ " following an MS name was used to denote the MS with different stereospecificity and ring structure from the common form (Table 3: SRS3). For example, while " G " represents D-glucopyranose, " G~ " represents L-glucofuranose, which is rarely seen. To represent reaction rules, " ~ " was used, instead, to convey logical negation [7,10,13,32]. For example, a constraint "~Ab" means the reaction will not happen if a β-galactose is present.

Ambiguous symbol 2 – " | ". For reaction rules, " | " is widely used to represent a potential branched structure in a substrate [7,10,13,32]. For example, "(GNb2|Ma3" represents a glycan structure with a potential branch on the mannose. However, " | " was originally designed to separate the certain and uncertain parts in a fragmented glycan, where there is a possibility of different structures (UR6).

Ambiguous symbol 3 – “ # ”. Originally, “ # ” was designated to signify the starting point of glycosides that are not amino acids or lipid moieties (Table 3: GR1). For example, “GNb3Ab” connected to a “4-trifluoroacetamidophenol” is written as “GNb3Ab#4-Trifluoroacetamidophenol.”

Ambiguous symbol 4 – “ * ”. Another ambiguous symbol is the asterisk “ * ”. In the original Linear Code context, “ * ” is used when an entire saccharide unit in the complex carbohydrate is unknown. In reaction rules representation, “ * ” marks the “reaction site”, the position of the first difference between product and substrate strings in Linear Code form [7,10,13,32]. Note that the “reaction site” does not necessarily refer to the exact place that the reaction happens. For example, given the reaction “(...Ab4GNb → (Fa3(...Ab4)GNb,” the constraint “ (*Ab4 or (*Fa2Ab4” means that the reaction will happen if and only if the “ … ” in the reactant represents either nothing or “Fa2.” In this case, “ * ” on the left of “Ab4” indicates where the reactant and the product differ from left to right in the Linear Code expression. However, the real reaction takes place at the “GNb,” not “Ab4.” Demonstrating the left-to-right specificity of “ * ”, consider the rule, (Ma2Ma → (Ma with constraint ~*2Ma3(…Ma6)Ma6. This constraint rules out removing the Ma2 on the middle branch (underlined) of the original M9 glycan, Ma2Ma2Ma3(<u>Ma2Ma3</u>(Ma2Ma6)Ma6)Mb4GNb4GN;Asn. If parsed from right to left, the constraint would be ~*Ma3(…Ma6)Ma6.

Linear Code is primarily a representation of glycan structure, and the formulation of reaction rules from Linear Code emerged as it was adapted for use with systems biology reaction networks. Specifically, when researchers aimed to define rules for reactions when building the networks, additional symbols were needed and, therefore, proposed. However, these now differ between studies.

In the first study, building reaction networks from Linear Code, Krambeck et al. defined “ … ”, “ _ ”, and “ | ” as uncertainty operators to indicate specific combinations or balanced or unbalanced (complete or incomplete) branches [7,10,32]. Spahn et al. used only two of the three symbols; “ | ” to indicate branching and “ … ” to represent continuation [13]. In this section, we will only focus on Krambeck et al. syntax. Syntactically, each of these symbols specifies whether or not the monosaccharides following the symbol, the first monosaccharide within the uncertainty operator replacement, appear within parentheses. If the monosaccharides appear within parentheses, it is “branching” off the lowest-carbon-index chain; otherwise, it is a “continuation” along the lowest-carbon-index chain. Each uncertainty operator describes a branching and/or continuation. Additionally, an uncertainty operator can require a complete phrase, with matched parentheses, or not. Finally, some uncertainty operators can be replaced with nothing (the empty string).

In the original Krambeck et al. implementation, multiple disjunctive constraints are connected by the logical disjunction “or.” An example is “(*Ab4 or (*NNa3Ab4” (Table 7). In the Liang et al. adaptation, however, the “or” relationship is delineated by writing each reaction rule on separate lines. For example, the two constraints for the reaction rule “(...Ab4GNb → (Fa3(...Ab4)GNb” would simply be written on two lines (Table 8).

Table 7: The reaction rule (GN → (Ab4GN with four constraints written in the same cell.

Enzyme	Reactant	Product	Constraint
b4GalT	(GN	(Ab4GN	*...GNb2|Ma3 or *...GNb4|Ma3 or *...GNb2|Ma6 or *...GNb6|Ma6

Table 8: The reaction rule (GN → (Ab4GN with four constraints written on separate lines.

Enzyme	Reactant	Product	Constraint
b4GalT	(GN	(Ab4GN	*...GNb2|Ma3
b4GalT	(GN	(Ab4GN	*...GNb4|Ma3
b4GalT	(GN	(Ab4GN	*...GNb2|Ma6
b4GalT	(GN	(Ab4GN	*...GNb6|Ma6

Most adaptations of reaction rule implementations are more or less related to the earliest Krambeck et al. adaptation. Some symbols are only seen in the Krambeck et al. adaptation. Besides the “ # ” as the number symbol, Krambeck et al. also uses “Gnbis” to refer to the specific structure of bisecting GN, which is “Ma3(GNb4)(...Ma6)Mb4.”

Several reaction rules for N-glycan biosynthesis are presented for direct comparison (Table 9, Table S5 in Supporting Information File 1). While there were several apparent divergences in the usage of terms, the rules are predominantly similar. The intent of this paper is to ensure the consistency of these rulesets going forward.

Table 9: Reaction rules from multiple N-glycan biosynthesis models in LiCoRR representation. This table describes select rules from Krambeck et al. [10] in LiCoRR and LiCoRRICE representation. Representations across multiple manuscripts can be found in Linear Code, LiCoRR and LiCoRRICE in Table S5 (Supporting Information File 1).

Enz.	Substrate	Product	Constraints (LiCoRR)	Constraints (LiCoRRICE)
ManI	(Ma2Ma	(Ma	!@2Ma3(…Ma6)Ma6 & !Ga3	nMan(a1-?)>4 & nMan(a1-?)<8 & !Man(a1-2)Man(a1-3)... Man(a1-6) & !Glc(a1-3)
ManI	(Ma3(Ma2Ma3(Ma6)Ma6)	(Ma3(Ma3(Ma6)Ma6)	!Ga3	!Glc(a1-3)
ManII	(Ma3(Ma6)Ma6	(Ma6Ma6	(GNb2+Ma3 & !Gnbis	!Gal(b1-?) & !GlcNAc(b1-4)...Man(b1-4) & GlcNAc(b1-2)Man(a1-3)
ManII	(Ma6Ma6	(Ma6	(GNb2+Ma3 & !Gnbis	
a6FucT	GNb4GN	GNb4(Fa6)GN	GNb2+Ma3 & #A=0 & !Gnbis	GlcNAc(b1-2)Man(a1-3)... Man(b1-4) & !GlcNAc(b1-4)...Man(b1-4) & !Fuc(a1-3)
GnTI	(Ma3(Ma3(Ma6)Ma6)Mb4	(GNb2Ma3(Ma3(Ma6)Ma6)Mb4		nMan(a1-?)=4
GnTII	(GNb2+Ma3(Ma6)Mb4	(GNb2+Ma3(GNb2Ma6)Mb4		nMan(a1-?)=2 & !GlcNAc(b1-4)...Man(b1-4) & !Fuc(a1-3) & !Gal(b1-?)
GnTIII	GNb2+Ma3	GNb2+Ma3(GNb4)	!Ab & !Gnbis	GlcNAc(b1-2)Man(a1-3)... Man(b1-4) & !Gal(b1-?)
GnTIV	(GNb2Ma3	(GNb2(GNb4)Ma3	!Gnbis	!Gal(b1-?) & !GlcNAc(b1-4)...Man(b1-4)
GnTV	(GNb2Ma6	(GNb2(GNb6)Ma6	!Gnbis	!Gal(b1-?) & !GlcNAc(b1-4)...Man(b1-4)
iGnT	(Ab4GN	(GNb3Ab4GN	!@_Ma3+Mb4	
b4GalT	(GN	(Ab4GN	!@GNb4)(...Ma6)Mb4	!Gal(b1-3)GlcNAc(b1-?) & !@GlcNAc(b1-4)...Man(b1-4)
b3GalT	(GN	(Ab3GN	!@GNb4)(...Ma6)Mb4	!Gal(b1-4)GlcNAc(b1-?) & !@GlcNAc(b1-4)...Man(b1-4)

Recommendations to Unify Descriptive Usages of Linear Code for Reaction Rules (*LiCoRR*)

Linear Code has shown its utility for the compact description of glycans and compatibility with efforts to define glycan reaction rules for systems biology models. A few ambiguities have emerged through different interpretations and implementations. Here we propose possible solutions as described by the original prescription for Linear Code, the consensus of the community, and our recommendation following this survey.

We have demonstrated the LiCoRR representation of all N-glycosylation reaction rules discussed in this paper in Table 9. Table 9 also includes an instance of these reaction rules written with IUPAC monosaccharides and linkages from GlycoEnzDB. Due to incomplete adoption and flexibility of Linear Code monosaccharides, we encourage users to accommodate both Linear Code and IUPAC monosaccharides when possible to facilitate interoperability; Linear Code monosaccharides may not be sufficient for every project while IUPAC-extended nomenclature [18] is actively maintained to ensure complete coverage of known sugars. If a user wants to specify that they are using LiCoRR with IUPAC monosaccharides, they can specify it as “LiCoRRICE” the LiCoRR-IUPAC Complement Expression. We also provide the matched constraints in Table 9 as Original Linear Code (Table S5 in Supporting Information File 1). It should be noted that IUPAC uses square brackets, "[]", rather than parentheses, "()", to delineate branching. Therefore, the wildcards should recognize square brackets rather than parentheses. Additionally, IUPAC does not use deterministic branching. Therefore, specifying branch direction is not meaningful and the three branch-specific LiCoRR wildcards can be reduced to one, "...", in LiCoRRICE. With these small changes, LiCoRR can be extended to LiCoRRICE and, as such, gain access to its carefully curated and growing list of MS units and modifications.

The original Linear Code syntax contains eighteen specific regulations across seven categories, among which only five regulations are seen in reaction rule implementations. In fact, the five regulations include three SU elements (MS name, linkage-type, position number), denotations (Table 3: SU) and one branch rule (Table 3: BR1). BR1 dictates that when two branching MSs are identical, the MS linked to the higher index carbon will have its chain on the branch (Table 3: BR1). If we extend the condition for BR1 from identical MSs to all MSs, written glycan structures will still maintain their uniqueness since each position on the MS can only connect to a single MS. BR2 dictates that the least frequent MS of the pair will branch (Table 3: BR2). BR2 solves the case when there are more than two non-reducing MSs linked to the same reducing MS. However, if we applied the expanded BR1 and ordered the chains based on decreasing position numbers from right to left in multi-chain cases, BR2 would be redundant. For example, Ab4(GNb4GNb3)(GNb6)Ab4Gb will be written as GNb4GNb3(Ab4)(GNb6)Ab4Gb.

Among the logical relationships required for constraint specification, only "or" is seen in the original Linear Code rules. " / " was designed to separate two possibilities within an SU (Table 3: UR4) and " // " was used to separate two possible complete SU options (Table 3: UR5). It would cause unnecessary confusion if " / " and " // " are used to denote the "or" relationship between constraints. Therefore, the task to convey Boolean logic among constraints was left to emerge organically in its application to reaction rules.

Recommendation 1 – "Logical negation." The field chose to use the " ~ " to indicate logical negation (Table 10: a). Unfortunately, this choice conflicts with the ability to express uncommon stereospecificity, as prescribed in the original Linear Code (Table 3: SRS3). Though this is a rare necessity, and the original Linear Code tilde appears on the right of the monosaccharide, usage of a " ! " - as used in many common programming languages – to indicate logical negation would preserve the original meaning of the tilde in case it becomes necessary in a future notation.

Recommendation 2 – "And." Similarly, the field chose " & " to represent the conjunction relationship between constraints. We recommended preserving this symbol use since it is human and computer-readable and does not overlap with any notation in the original Linear Code.

Recommendation 3 – "Number." " # " was defined to combine glycans with glycosides other than amino acids and lipids (Table 3: GR1). Krambeck et al. use it to represent the number of times a certain MS appears, a common use of " # ". In LiCoRR, we deprecate the use of " # ", " ; ", and " : " to specify the glycoconjugate class. The number sigh " # " can be used to separate a glycan (on the left) from any conjugate (on the right). Colon and semicolon can therefore be reserved for other future uses. To specify a glycopeptide, users may also inscribe them directly in the peptide using the existing branching rules: "PEP(AG(LY)CAN)TIDE" would describe a biantennary glycan bound to the threonine of a peptide. Because the number sign is used to indicate a glycoconjugate, we recommend using "n." For example, "#A = 5" will then be written as "nA=5" (Table 10: i).

Recommendation 4 – "Splitting & 'or'." In addition to having several constraints split by "or," we can rewrite the rules several times with a single constraint for each rule, as done for the reaction rule b4GalT in [14]. Splitting disjunctions over multiple lines is similar to atomization, the first normal form of database normalization requiring the domain of each attribute to contain an indivisible element. In addition, the separate rules have the advantage that they can have different reaction rate parameters. This advantage can eliminate the need for separate adjustment rules for various cases. Depending on the circumstances, splitting disjunctions across multiple lines may be necessary, though it is often more succinct to condense them, separated by an "or" within a single rule.

Recommendation 5 – "Branch point." Many studies using Linear Code to define glycan synthesis networks assigned " | " as a possible branch point [7,10,13]. Our recommendation, however, is to use " + " instead of " | " as the branch point because " | " is already assigned within the original Linear Code. Additionally, we think " + " is more morphologically close to a branch.

Recommendation 6 – "Omission." Though " * " has been widely used by the systems glycobiology field to represent the reaction site, the original Linear Code rules actually specify " * " to stand for the omission of an entire saccharide unit (Table 3: UR3). We wish to minimize this inconsistency with the original statement of Linear Code [31]. Therefore, for " * ", we recommend preserving the meaning of the omission of one

Table 10: Symbols previously used by systems glycobiologists and our recommendations. Rows a–i are the functions implemented by published papers. Rows j–m are the functions prescribed in the original Linear Code rules. (A) Symbols to represent reaction rules across publications utilizing Linear Code. (B) Consensus and recommendation for reaction rule representation going forward.

	(A)						(B)		
	Symbol used	OLC [31]	Kra [10]	Spa [13]	Lia [14]	Hou [7]	Consensus adaptation of OLC to reaction rules	LiCoRR	Examples
a	logical negation		~	~	~	~	~	!	!Ma
b	and		&			&	&	&	!Ma & Ab3
c	or		or			or	or	separate rules, or	!Ma or Ab3
d	continuation (left parenthesis matched to right parenthesis.)		_	...	...	_	... or _	_	see Table 6
e	ligand (all parenthesis matched)		...			...	...	...	see Table 6
f	possible branch point		|	|	|	|	|	+	see Table 6
g	reaction site (Code change site)		*	*	*	*	*	@	!@…Ma2
h	possible modification						$	$	A$GN
i	number		#				#	n	nA=0 nA>2
j	divide certainty and uncertainty (Table 2: UR6)	|					nothing	nothing	
k	omission of an entire SU (Table 2: UR3)	*					nothing	*	ANb3*N
l	glycosides (Table 2: GR1)	;, :, #		;			nothing	; for amino acid, : for lipid moieties, # for other glycosides	Ga;NY-S-C Gb:C
m	MS with uncommon stereospecificity and ring structure (Table 2: SRS3)	~					nothing	~	L-Glcf: G˜

Abbreviations: OLC (Original Linear Code [31]), Kra (Krambeck et al. [10]), Spa (Spahn [13]), Lia (Liang et al. [14]), Hou (Hou et al. [7]).

entire SU. In theory, according to Banin's definition, a saccharide unit can be specified as " ??? ". Using " * " to indicate a complete SU, would avoid using an unmanageable number of question marks to represent an ambiguous glycan. Question marks should still be used to indicate unknown elements of an SU (e.g., "Ab4Gb" without knowledge of "b4G" could be written as "A???b"), but there should never be four adjacent question marks. We propose a substitute for the reaction site in Recommendation 7.

Recommendation 7 – "Reaction site." The reaction site is the location of the first change to the glycan expression. Because " * " is already defined within Linear Code to indicate "omission," we choose " @ " to indicate the reaction site. The reaction site, in previous reaction rules as " * " and going forward as " @ ", is the position of the first difference between product and substrate strings in the Linear Code form.

Recommendation 8 – "Modification." As specified in the original Linear Code, we recommend using " [] " to represent known modifications (Table 3: MR1). For example, "A[2P]" represents a galactose with its second position modified by a phosphate. However, this specific modification may not always be known. Therefore, in addition to " [] " as exact modifications, we recommended using the " $ " sign to represent a possible modification site. For example, "A$GN" represents a

GlcNAc connected to a galactose that might be modified. The modification can be specified (e.g., phosphorylation on the 2nd carbon) in the typical way, with square brackets “A$[2P]GN”.

Recommendation 9 – “Branching index.” In LiCoRR we have deprecated the original linear code branching rules due to redundancy and default to a version of BR1: Regardless of whether the MSs are equivalent, the MS linked to the higher index carbon will branch (appear first in the written representation when read right to left, reducing to non-reducing end). This rule can be extended to glycopeptides providing a means of representing glycans directly embedded in a glycopeptide. “PEP(Gal[3S]b3(GNb6)AN)TIDE” would describe a trisaccharide O-glycan bound to the threonine of an eponymously named glycoprotein.

Overall, the consensus in these representations centers around the foundational work of the original Linear Code paper [31] and Krambeck et al. [10]. We have simply highlighted gaps in clarity that have resulted in colloquially small but computationally important divergences throughout the literature.

Conclusion

The field of systems glycobiology is poised to tackle increasingly complex glycan synthesis problems owing to the advent of a number of enabling computational modeling technologies. Linear Code is used to represent reaction rules of glycan synthesis thereby bringing both human-readability and computer-parsability to the glycoinformatics space. The utility of Linear Code in glycoinformatics has been extended by the inclusion of new symbols, relations, and attributes that accommodate the challenge of specifying reaction rules. Yet various implementations conflict with each other and the original Linear Code. Here, we have delineated the various adaptations made to accommodate reaction rule representation, the discordance between various implementations, and proposed a consensus for future representations called LiCoRR.

The adoption of a common reaction rule representation would increase FAIR (Findable, Accessible, Interoperable, Reusable) standards [40] compliance in glycoinformatics which will have far-reaching implications. As demonstrated by WURCS, a deterministic exemplar of glycan representation that can be used as a database key, “findability” can be improved by unifying data with metadata. While not fully deterministic, LiCoRR is a predictable representation for reaction rules thereby findability search through data-metadata unification. Towards improving the findability of glycans through data-metadata unification, we provide a parser (gRegex, see Supporting Information File 1) and a context-free grammar which should facilitate integration into several formal-language compatible glycoinformatics tools including glycologue [41], glypy [42], glycome-db [43]; adoption LiCoRR or these wildcards in other glycan representations could shift glycan-database search from monosaccharide count to substructure class specification. While computational tools exist to compare XML-type models directly [44], the verbosity of the models can challenge comprehension. While less descriptive, succinct human-readable and understandable LiCoRR expressions provide an opportunity for a human observer to manually compare and consider two related models. Ideally, succinct, readable, and comprehensible reaction rules sets will be sufficiently standardized, like XML-type representations, so that they will be “interoperable” across multiple modeling software so that models can be “reused,” reproduced, validated, and extended across labs. Toward encouraging the reuse of LiCoRR, we would like to acknowledge the trademark held by a former company, Glycominds Ltd. As our work is an extension and consolidation of novel development throughout the public domain, and we have no intent to exploit the trademark for financial gain, it is our understanding that we may publish freely and dedicate LiCoRR to the public domain under a CC-BY free-use with attribution license. Increased readability and FAIRness through clarifying the nomenclature will help advance glycoinformatics technologies by making possible cross-platform and multi-omics integration and interpretation; interoperability may be enhanced through a community-endorsed vocabulary.

We further hope that the symbols described in this work, specifically the wildcards, will be used in other glycan representations and applications beyond biosynthesis modeling. The definition of glycan classes can be useful for efficiently and unambiguously describing the key elements of large complex glycans while only communicating the central information. Adoption of these symbols, now well-defined symbols, by more popular representations, such as IUPAC, could increase both the flexibility and succinctness of those representations. We believe the utility of these wild-cards extends beyond biosynthesis modeling (Table 9) and may be useful in the description of glycan-chemosynthetic procedures, lectin identification of glycan motifs, and any other purpose where a group of glycans (rather than an individual glycan) is being discussed or described. We hope to encourage that adoption through our LiCoRRICE examples.

Increased FAIRness will facilitate the validation and distribution of developing glycoinformatics toolkits. Easy-to-use glycoinformatics toolkits, made possible by the fluency of interoperability across tools, are one mechanism by which glycobiology can be shared with the broader community of biology.

Supporting Information

Supporting Information File 1

Supporting tables.
[https://www.beilstein-journals.org/bjoc/content/supplementary/1860-5397-16-215-S1.zip]

Funding

This work was supported by generous funding from the Novo Nordisk Foundation provided to the Center for Biosustainability at the Technical University of Denmark (NNF10CC1016517) and from NIGMS (R35 GM119850). Additional funding was provided by the Triton Research & Experiential Learning Scholars (TRELS) program at the University of California, San Diego, the NIDDK Information Network (dkNET) supported by NIH & NIDDK (2U24DK097771-06), and NIH (NIH HL103411).

Authors

Benjamin P. Kellman[1,2,3], Yujie Zhang[1,2], Emma Logomasini[1,2], Eric Meinhardt[4], Karla P. Godinez-Macias[1], Austin W. T. Chiang[1,2], James T. Sorrentino[1,2,3], Chenguang Liang[1,2], Bokan Bao[1,2], Yusen Zhou[5], Sachiko Akase[6], Isami Sogabe[6], Thukaa Kouka[6], Elizabeth A Winzeler[1], Iain B. H. Wilson[7,8], Matthew P. Campbell[7,9], Sriram Neelamegham[5,7], Frederick J. Krambeck[7,10,11], Kiyoko F. Aoki-Kinoshita[6,7,12] and Nathan E. Lewis[*,1,2,3,7,13]

Addresses

[1]Department of Pediatrics, University of California San Diego School of Medicine, 9500 Gilman Dr, La Jolla, 92093, California, USA, [2]Department of Bioengineering, University of California San Diego School of Engineering, 9500 Gilman Dr, La Jolla, 92093, California, USA, [3]Bioinformatics and Systems Biology Program, University of California San Diego School of Engineering, 9500 Gilman Dr, La Jolla, 92093, California, USA, [4]Department of Linguistics, University of California San Diego, 9500 Gilman Dr, La Jolla, 92093, California, USA, [5]Department of Chemical and Biological Engineering School of Engineering and Applied Sciences, State University of New York, University at Buffalo, 303 Furnas Hall, Buffalo, 14260-4200, New York, USA, [6]Graduate School of Engineering, Soka University, 1-236 Tangi-machi, Hachioji-shi, Tokyo, Japan, [7]Systems Glycobiology Consortium, [8]Institute of Biochemistry, University of Natural Resources and Life Sciences, Gregor-Mendel-Straße 33, 1180 Vienna, Austria, [9]Institute for Glycomics, Griffith University, Glycomics 1, G26/1 Parklands Dr, Southport QLD 4215, Australia, [10]Department of Chemical and Biomolecular Engineering, Johns Hopkins University, 3400 North Charles Street, Baltimore, 21218, Maryland, USA, [11]ReacTech Inc., 810 Cameron St, Alexandria, 22314, Virginia, USA, [12]Faculty of Science and Engineering, Soka University, 1-236 Tangi-machi, Hachioji-shi, Tokyo, Japan, and [13]Novo Nordisk Foundation Center for Biosustainability at the University of California San Diego School of Medicine, 9500 Gilman Dr. La Jolla, 92093, California, USA.

ORCID® iDs

Benjamin P. Kellman - https://orcid.org/0000-0002-0780-6096
Yujie Zhang - https://orcid.org/0000-0002-1017-0316
Thukaa Kouka - https://orcid.org/0000-0002-4442-2294
Iain B. H. Wilson - https://orcid.org/0000-0001-8996-1518
Matthew P. Campbell - https://orcid.org/0000-0002-9525-792X
Frederick J. Krambeck - https://orcid.org/0000-0003-3838-3775
Nathan E. Lewis - https://orcid.org/0000-0001-7700-3654

Preprint

A non-peer-reviewed version of this article has been previously published as a preprint: doi:10.1101/2020.05.31.126623

References

1. Kontoravdi, C.; Jimenez del Val, I. *Curr. Opin. Chem. Eng.* **2018,** *22,* 89–97. doi:10.1016/j.coche.2018.08.007
2. Spahn, P. N.; Lewis, N. E. *Curr. Opin. Biotechnol.* **2014,** *30,* 218–224. doi:10.1016/j.copbio.2014.08.004
3. Puri, A.; Neelamegham, S. *Ann. Biomed. Eng.* **2012,** *40,* 816–827. doi:10.1007/s10439-011-0464-5
4. Krambeck, F. J.; Bennun, S. V.; Andersen, M. R.; Betenbaugh, M. J. *PLoS One* **2017,** *12,* e0175376. doi:10.1371/journal.pone.0175376
5. Sou, S. N.; Jedrzejewski, P. M.; Lee, K.; Sellick, C.; Polizzi, K. M.; Kontoravdi, C. *Biotechnol. Bioeng.* **2017,** *114,* 1570–1582. doi:10.1002/bit.26225
6. del Val, I. J.; Polizzi, K. M.; Kontoravdi, C. *Sci. Rep.* **2016,** *6,* 28547. doi:10.1038/srep28547
7. Hou, W.; Qiu, Y.; Hashimoto, N.; Ching, W.-K.; Aoki-Kinoshita, K. F. *BMC Bioinf.* **2016,** *17* (Suppl. 7), 240. doi:10.1186/s12859-016-1094-6
8. Liu, G.; Neelamegham, S. *Wiley Interdiscip. Rev.: Syst. Biol. Med.* **2015,** *7,* 163–181. doi:10.1002/wsbm.1296
9. McDonald, A. G.; Hayes, J. M.; Bezak, T.; Głuchowska, S. A.; Cosgrave, E. F. J.; Struwe, W. B.; Stroop, C. J. M.; Kok, H.; van de Laar, T.; Rudd, P. M.; Tipton, K. F.; Davey, G. P. *J. Cell Sci.* **2014,** *127,* 5014–5026. doi:10.1242/jcs.151878
10. Krambeck, F. J.; Bennun, S. V.; Narang, S.; Choi, S.; Yarema, K. J.; Betenbaugh, M. J. *Glycobiology* **2009,** *19,* 1163–1175. doi:10.1093/glycob/cwp081
11. Liu, G.; Marathe, D. D.; Matta, K. L.; Neelamegham, S. *Bioinformatics* **2008,** *24,* 2740–2747. doi:10.1093/bioinformatics/btn515
12. Liu, G.; Puri, A.; Neelamegham, S. *Bioinformatics* **2013,** *29,* 404–406. doi:10.1093/bioinformatics/bts703
13. Spahn, P. N.; Hansen, A. H.; Hansen, H. G.; Arnsdorf, J.; Kildegaard, H. F.; Lewis, N. E. *Metab. Eng.* **2016,** *33,* 52–66. doi:10.1016/j.ymben.2015.10.007
14. Liang, C.; Chiang, A. W. T.; Hansen, A. H.; Arnsdorf, J.; Schoffelen, S.; Sorrentino, J. T.; Kellman, B. P.; Bao, B.; Voldborg, B. G.; Lewis, N. E. *Curr. Res. Biotechnol.* **2020,** *2,* 22–36. doi:10.1016/j.crbiot.2020.01.001

15. Spahn, P. N.; Hansen, A. H.; Kol, S.; Voldborg, B. G.; Lewis, N. E. *Biotechnol. J.* **2017,** *12,* 1600489. doi:10.1002/biot.201600489
16. McDonald, A. G.; Tipton, K. F.; Davey, G. P. *PLoS Comput. Biol.* **2016,** *12,* e1004844. doi:10.1371/journal.pcbi.1004844
17. Bao, B.; Kellman, B. P.; Chiang, A. W. T.; York, A. K.; Mohammad, M. A.; Haymond, M. W.; Bode, L.; Lewis, N. E. *bioRxiv* **2019,** 693507. doi:10.1101/693507
18. Neelamegham, S.; Aoki-Kinoshita, K.; Bolton, E.; Frank, M.; Lisacek, F.; Lütteke, T.; O'Boyle, N.; Packer, N. H.; Stanley, P.; Toukach, P.; Varki, A.; Woods, R. J.; The SNFG Discussion Group. *Glycobiology* **2019,** *29,* 620–624. doi:10.1093/glycob/cwz045
19. Mehta, A. Y.; Cummings, R. D. *Bioinformatics* **2020,** *36,* 3613–3614. doi:10.1093/bioinformatics/btaa190
20. Cheng, K.; Zhou, Y.; Neelamegham, S. *Glycobiology* **2017,** *27,* 200–205. doi:10.1093/glycob/cww115
21. Demir, E.; Cary, M. P.; Paley, S.; Fukuda, K.; Lemer, C.; Vastrik, I.; Wu, G.; D'Eustachio, P.; Schaefer, C.; Luciano, J.; Schacherer, F.; Martinez-Flores, I.; Hu, Z.; Jimenez-Jacinto, V.; Joshi-Tope, G.; Kandasamy, K.; Lopez-Fuentes, A. C.; Mi, H.; Pichler, E.; Rodchenkov, I.; Splendiani, A.; Tkachev, S.; Zucker, J.; Gopinath, G.; Rajasimha, H.; Ramakrishnan, R.; Shah, I.; Syed, M.; Anwar, N.; Babur, Ö.; Blinov, M.; Brauner, E.; Corwin, D.; Donaldson, S.; Gibbons, F.; Goldberg, R.; Hornbeck, P.; Luna, A.; Murray-Rust, P.; Neumann, E.; Ruebenacker, O.; Samwald, M.; van Iersel, M.; Wimalaratne, S.; Allen, K.; Braun, B.; Whirl-Carrillo, M.; Cheung, K.-H.; Dahlquist, K.; Finney, A.; Gillespie, M.; Glass, E.; Gong, L.; Haw, R.; Honig, M.; Hubaut, O.; Kane, D.; Krupa, S.; Kutmon, M.; Leonard, J.; Marks, D.; Merberg, D.; Petri, V.; Pico, A.; Ravenscroft, D.; Ren, L.; Shah, N.; Sunshine, M.; Tang, R.; Whaley, R.; Letovksy, S.; Buetow, K. H.; Rzhetsky, A.; Schachter, V.; Sobral, B. S.; Dogrusoz, U.; McWeeney, S.; Aladjem, M.; Birney, E.; Collado-Vides, J.; Goto, S.; Hucka, M.; Le Novère, N.; Maltsev, N.; Pandey, A.; Thomas, P.; Wingender, E.; Karp, P. D.; Sander, C.; Bader, G. D. *Nat. Biotechnol.* **2010,** *28,* 935–942. doi:10.1038/nbt.1666
22. Lloyd, C. M.; Halstead, M. D. B.; Nielsen, P. F. *Prog. Biophys. Mol. Biol.* **2004,** *85,* 433–450. doi:10.1016/j.pbiomolbio.2004.01.004
23. Herget, S.; Ranzinger, R.; Maass, K.; Lieth, C.-W. v. d. *Carbohydr. Res.* **2008,** *343,* 2162–2171. doi:10.1016/j.carres.2008.03.011
24. Tanaka, K.; Aoki-Kinoshita, K. F.; Kotera, M.; Sawaki, H.; Tsuchiya, S.; Fujita, N.; Shikanai, T.; Kato, M.; Kawano, S.; Yamada, I.; Narimatsu, H. *J. Chem. Inf. Model.* **2014,** *54,* 1558–1566. doi:10.1021/ci400571e
25. Matsubara, M.; Aoki-Kinoshita, K. F.; Aoki, N. P.; Yamada, I.; Narimatsu, H. *J. Chem. Inf. Model.* **2017,** *57,* 632–637. doi:10.1021/acs.jcim.6b00650
26. Aoki-Kinoshita, K.; Agravat, S.; Aoki, N. P.; Arpinar, S.; Cummings, R. D.; Fujita, A.; Fujita, N.; Hart, G. M.; Haslam, S. M.; Kawasaki, T.; Matsubara, M.; Moreman, K. W.; Okuda, S.; Pierce, M.; Ranzinger, R.; Shikanai, T.; Shinmachi, D.; Solovieva, E.; Suzuki, Y.; Tsuchiya, S.; Yamada, I.; York, W. S.; Zaia, J.; Narimatsu, H. *Nucleic Acids Res.* **2016,** *44,* D1237–D1242. doi:10.1093/nar/gkv1041
27. Campbell, M. P.; Peterson, R.; Mariethoz, J.; Gasteiger, E.; Akune, Y.; Aoki-Kinoshita, K. F.; Lisacek, F.; Packer, N. H. *Nucleic Acids Res.* **2014,** *42,* D215–D221. doi:10.1093/nar/gkt1128
28. York, W. S.; Mazumder, R.; Ranzinger, R.; Edwards, N.; Kahsay, R.; Aoki-Kinoshita, K. F.; Campbell, M. P.; Cummings, R. D.; Feizi, T.; Martin, M.; Natale, D. A.; Packer, N. H.; Woods, R. J.; Agarwal, G.; Arpinar, S.; Bhat, S.; Blake, J.; Castro, L. J. G.; Fochtman, B.; Gildersleeve, J.; Goldman, R.; Holmes, X.; Jain, V.; Kulkarni, S.; Mahadik, R.; Mehta, A.; Mousavi, R.; Nakarakommula, S.; Navelkar, R.; Pattabiraman, N.; Pierce, M. J.; Ross, K.; Vasudev, P.; Vora, J.; Williamson, T.; Zhang, W. *Glycobiology* **2020,** *30,* 72–73. doi:10.1093/glycob/cwz080
29. Alocci, D.; Mariethoz, J.; Gastaldello, A.; Gasteiger, E.; Karlsson, N. G.; Kolarich, D.; Packer, N. H.; Lisacek, F. *J. Proteome Res.* **2019,** *18,* 664–677. doi:10.1021/acs.jproteome.8b00766
30. Panico, R.; Richer, J.-C., Eds. *A Guide to IUPAC Nomenclature of Organic Compounds: Recommendations 1993;* Blackwell Science, Inc., 1993.
31. Banin, E.; Neuberger, Y.; Altshuler, Y.; Halevi, A.; Inbar, O.; Nir, D.; Dukler, A. *Trends Glycosci. Glycotechnol.* **2002,** *14,* 127–137. doi:10.4052/tigg.14.127
32. Bennun, S. V.; Yarema, K. J.; Betenbaugh, M. J.; Krambeck, F. J. *PLoS Comput. Biol.* **2013,** *9,* e1002813. doi:10.1371/journal.pcbi.1002813
33. Varki, A.; Sherman, W.; Kornfeld, S. *Arch. Biochem. Biophys.* **1983,** *222,* 145–149. doi:10.1016/0003-9861(83)90511-8
34. Rohrer, J.; Kornfeld, R. *Mol. Biol. Cell* **2001,** *12,* 1623–1631. doi:10.1091/mbc.12.6.1623
35. Hare, N. J.; Lee, L. Y.; Loke, I.; Britton, W. J.; Saunders, B. M.; Thaysen-Andersen, M. *J. Proteome Res.* **2017,** *16,* 247–263. doi:10.1021/acs.jproteome.6b00685
36. Röttger, S.; White, J.; Wandall, H. H.; Olivo, J. C.; Stark, A.; Bennett, E. P.; Whitehouse, C.; Berger, E. G.; Clausen, H.; Nilsson, T. *J. Cell Sci.* **1998,** *111,* 45–60.
37. Akune, Y.; Lin, C.-H.; Abrahams, J. L.; Zhang, J.; Packer, N. H.; Aoki-Kinoshita, K. F.; Campbell, M. P. *Carbohydr. Res.* **2016,** *431,* 56–63. doi:10.1016/j.carres.2016.05.012
38. McDonald, A. G.; Boyce, S.; Tipton, K. F. *Nucleic Acids Res.* **2009,** *37,* D593–D597. doi:10.1093/nar/gkn582
39. Agravat, S. B.; Song, X.; Rojsajjakul, T.; Cummings, R. D.; Smith, D. F. *Bioinformatics* **2016,** *32,* 1471–1478. doi:10.1093/bioinformatics/btw048
40. Wilkinson, M. D.; Dumontier, M.; Aalbersberg, I. J.; Appleton, G.; Axton, M.; Baak, A.; Blomberg, N.; Boiten, J.-W.; da Silva Santos, L. B.; Bourne, P. E.; Bouwman, J.; Brookes, A. J.; Clark, T.; Crosas, M.; Dillo, I.; Dumon, O.; Edmunds, S.; Evelo, C. T.; Finkers, R.; Gonzalez-Beltran, A.; Gray, A. J. G.; Groth, P.; Goble, C.; Grethe, J. S.; Heringa, J.; 't Hoen, P. A. C.; Hooft, R.; Kuhn, T.; Kok, R.; Kok, J.; Lusher, S. J.; Martone, M. E.; Mons, A.; Packer, A. L.; Persson, B.; Rocca-Serra, P.; Roos, M.; van Schaik, R.; Sansone, S.-A.; Schultes, E.; Sengstag, T.; Slater, T.; Strawn, G.; Swertz, M. A.; Thompson, M.; van der Lei, J.; van Mulligen, E.; Velterop, J.; Waagmeester, A.; Wittenburg, P.; Wolstencroft, K.; Zhao, J.; Mons, B. *Sci. Data* **2016,** *3,* 160018. doi:10.1038/sdata.2016.18
41. McDonald, A. G.; Tipton, K. F.; Stroop, C. J.; Davey, G. P. *BMC Res. Notes* **2010,** *3,* 173. doi:10.1186/1756-0500-3-173
42. Klein, J.; Zaia, J. *J. Proteome Res.* **2019,** *18,* 3532–3537. doi:10.1021/acs.jproteome.9b00367
43. Alocci, D.; Mariethoz, J.; Horlacher, O.; Bolleman, J. T.; Campbell, M. P.; Lisacek, F. *PLoS One* **2015,** *10,* e0144578. doi:10.1371/journal.pone.0144578
44. Scharm, M.; Wolkenhauer, O.; Waltemath, D. *Bioinformatics* **2016,** *32,* 563–570. doi:10.1093/bioinformatics/btv484