



ROLLING SMARTS:  
YOU DON'T ALWAYS FIND WHAT YOU WANT,  
BUT IF YOU TRY SOMETIMES,  
YOU FIND WHAT YOU NEED

Roger Sayle and John Mayfield  
NextMove Software, Cambridge, UK



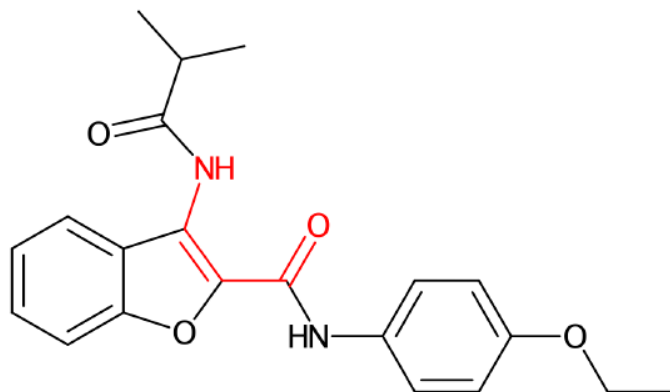
# FOR THE YOUNGER GENERATION...

- “We need a president with tremendous... SMARTS...”  
- Donald J. Trump



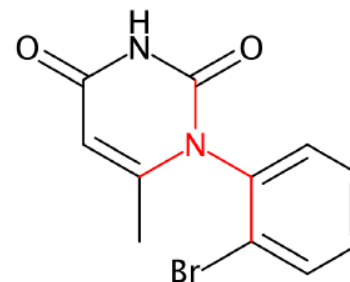
# WHAT IS SMARTS?

- SMARTS is a line notation for specifying substructural patterns in (or queries on) chemical substructures.
- SMARTS are to SMILES, as regular expressions (regex) are to strings, and Biovia query files are to Mol files.



O=[C,N]aa[N,O;!H0]

intramolecular H Bonds



[aD3]a!@-a([aD3])[aD3]

biaryl atropisomers



# SMARTS ATOM PRIMITIVES

- The portable subset of atom primitives include:

Symbol	Name	Definition	Default
<symbol>	(arom)element	IUPAC symbol (upper=A, lower=a)	(no default)
*	wildcard		(no default)
#n	atomic number	atomic number <n>	(no default)
A	aliphatic	aliphatic	(no default)
a	aromatic	aromatic	(no default)
R<0>	ring membership	in ring	any ring atom
X<n>	connectivity	<n> total connections	exactly one
D<n>	degree	<n> explicit connections	
H<n>	total H count	<n> attached hydrogens	
h<n>	Implicit H count	<n> implicit hydrogens	at least one
v<n>	valence	total bond order <n>	exactly one
x<n>	ring connectivity	<n> total ring connections	any ring atom
r<n>	smallest ring size	in smallest ring of size <n>	any ring atom
+<n>	positive charge	+<n> formal charge	+1 charge
-<n>	negative charge	-<n> formal charge	-1 charge



# SMARTS ATOM PRIMITIVE EXTENSIONS

- Additional atom primitives include:

Symbol	Definition	Toolkits
R<n>	Input order dependent SSSR membership	Daylight
R<n>	Symmetrized SSSR membership	RDKit
R<n>	Number of ring bonds	OpenEye (NextMove)
z<n>	Heteroatom neighbor count	RDKit, CACTVS
Z<n>	Aliphatic heteroatom neighbor count	RDKit, CACTVS
Z<n>	In Ring of size <n>	NextMove
G<n>	Group <n> element	CCG MOE
a<n>	<n> explicit connections	CACTVS, NextMove
i<0>	Saturated or unsaturated atom	Lilly, CACTVS, NextMove
^<n>	<n> attached hydrogens	RDKit, OpenEye

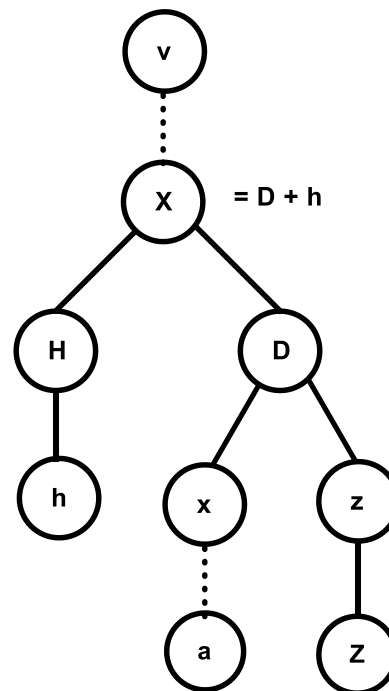
- Several toolkits support ranges: X{m-n}, X{m-}, X{-n}
- Several toolkits support bounds: X>m, X<n



# SMARTS ALGEBRA: $X = D + h$

- The most important axiom of SMARTS algebra is that  $X$ , the connectivity, is the degree ( $D$ ) plus the implicit hydrogen count ( $h$ ).

v	valence
X	connectivity
H	total hcount
h	implicit hcount
D	degree
x	ring bonds
a	aromatic bonds
z	hetero neighbors (RDKit)
Z	aliphatic hetero neighbors (RDKit)



# LOGICAL OPERATORS

- SMARTS primitives can be negated with ! prefix.
- Primitives can be combined by conjunction (AND) and disjunction (OR).
- The (often optional) & operator has the highest priority, and binds tighter than OR.
- Atom lists are implemented by the “,” disjunction operator, for example the familiar “[F,Cl,Br,I]”.
- The low priority AND operator “;” can be used for constraining disjunctive lists.



# DE MORGAN'S LAWS

- Arbitrarily complex Boolean expressions can be expressed in SMARTS, thanks to De Morgan's laws.
- e.g. The negation of "[F,Cl,Br,I]" is "[!F!Cl!Br!I]".
- The stable isotopes of hydrogen are "[0#1,1#1,2#1]", so unstable H isotopes "[#1;!0,!#1;!1,!#1;!2,!#1]".







# RADIOACTIVE SMARTS

[!0,!#1;!1,!#1;!2,!#1;!0,!#2;!3,!#2;!4,!#2;!0,!#3;!6,!#3;!7,!#3;!0,!#4;!9,!#4;!0,!#5;!10,!#5;!11,!#5;!0,!#6;!12,!#6;!13,!#6;!0,!#7;!14,!#7;!15,!#7;!0,!#8;!16,!#8;!17,!#8;!18,!#8;!0,!#9;!19,!#9;!0,!#10;!20,!#10;!21,!#10;!22,!#10;!0,!#11;!23,!#11;!0,!#12;!24,!#12;!25,!#12;!26,!#12;!0,!#13;!27,!#13;!0,!#14;!28,!#14;!29,!#14;!30,!#14;!0,!#15;!31,!#15;!0,!#16;!32,!#16;!33,!#16;!34,!#16;!36,!#16;!0,!#17;!35,!#17;!37,!#17;!0,!#18;!36,!#18;!38,!#18;!40,!#18;!0,!#19;!39,!#19;!41,!#19;!0,!#20;!40,!#20;!42,!#20;!43,!#20;!44,!#20;!46,!#20;!0,!#21;!47,!#21;!0,!#22;!46,!#22;!47,!#22;!48,!#22;!49,!#22;!50,!#22;!0,!#23;!51,!#23;!0,!#24;!50,!#24;!52,!#24;!53,!#24;!54,!#24;!0,!#25;!55,!#25;!0,!#26;!54,!#26;!56,!#26;!57,!#26;!58,!#26;!0,!#27;!59,!#27;!0,!#28;!58,!#28;!60,!#28;!61,!#28;!62,!#28;!64,!#28;!0,!#29;!63,!#29;!65,!#29;!0,!#30;!64,!#30;!66,!#30;!67,!#30;!68,!#30;!70,!#30;!0,!#31;!69,!#31;!71,!#31;!0,!#32;!70,!#32;!72,!#32;!73,!#32;!74,!#32;!0,!#33;!75,!#33;!0,!#34;!74,!#34;!76,!#34;!77,!#34;!78,!#34;!80,!#34;!0,!#35;!79,!#35;!81,!#35;!0,!#36;!79,!#36;!80,!#36;!82,!#36;!83,!#36;!84,!#36;!86,!#36;!0,!#37;!85,!#37;!0,!#38;!84,!#38;!86,!#38;!87,!#38;!88,!#38;!0,!#39;!89,!#39;!0,!#40;!90,!#40;!91,!#40;!92,!#40;!94,!#40;!96,!#40;!0,!#41;!93,!#41;!0,!#42;!92,!#42;!94,!#42;!95,!#42;!96,!#42;!97,!#42;!98,!#42;!0,!#44;!96,!#44;!98,!#44;!99,!#44;!100,!#44;!101,!#44;!102,!#44;!104,!#44;!0,!#45;!103,!#45;!0,!#46;!102,!#46;!104,!#46;!105,!#46;!106,!#46;!108,!#46;!110,!#46;!0,!#47;!107,!#47;!109,!#47;!0,!#48;!106,!#48;!108,!#48;!110,!#48;!111,!#48;!112,!#48;!114,!#48;!0,!#49;!113,!#49;!0,!#50;!112,!#50;!114,!#50;!115,!#50;!116,!#50;!117,!#50;!118,!#50;!119,!#50;!120,!#50;!122,!#50;!124,!#50;!0,!#51;!121,!#51;!123,!#51;!0,!#52;!120,!#52;!122,!#52;!123,!#52;!124,!#52;!125,!#52;!126,!#52;!0,!#53;!127,!#53;!0,!#54;!124,!#54;!126,!#54;!128,!#54;!129,!#54;!130,!#54;!131,!#54;!132,!#54;!134,!#54;!136,!#54;!0,!#55;!133,!#55;!0,!#56;!130,!#56;!132,!#56;!134,!#56;!135,!#56;!136,!#56;!137,!#56;!138,!#56;!0,!#57;!139,!#57;!0,!#58;!136,!#58;!138,!#58;!140,!#58;!142,!#58;!0,!#59;!141,!#59;!0,!#60;!142,!#60;!143,!#60;!145,!#60;!146,!#60;!148,!#60;!0,!#62;!144,!#62;!149,!#62;!150,!#62;!152,!#62;!154,!#62;!0,!#63;!153,!#63;!0,!#64;!154,!#64;!155,!#64;!156,!#64;!157,!#64;!158,!#64;!160,!#64;!0,!#65;!159,!#65;!0,!#66;!156,!#66;!158,!#66;!160,!#66;!161,!#66;!162,!#66;!163,!#66;!164,!#66;!0,!#67;!165,!#67;!0,!#68;!162,!#68;!164,!#68;!166,!#68;!167,!#68;!168,!#68;!170,!#68;!0,!#69;!169,!#69;!0,!#70;!168,!#70;!170,!#70;!171,!#70;!172,!#70;!173,!#70;!174,!#70;!176,!#70;!0,!#71;!175,!#71;!0,!#72;!176,!#72;!177,!#72;!178,!#72;!179,!#72;!180,!#72;!0,!#73;!180,!#73;!181,!#73;!0,!#74;!182,!#74;!183,!#74;!184,!#74;!186,!#74;!0,!#75;!185,!#75;!0,!#76;!184,!#76;!187,!#76;!188,!#76;!189,!#76;!190,!#76;!192,!#76;!0,!#77;!191,!#77;!193,!#77;!0,!#78;!192,!#78;!194,!#78;!195,!#78;!196,!#78;!198,!#78;!0,!#79;!197,!#79;!0,!#80;!196,!#80;!198,!#80;!199,!#80;!200,!#80;!201,!#80;!202,!#80;!204,!#80;!0,!#81;!203,!#81;!205,!#81;!0,!#82;!204,!#82;!206,!#82;!207,!#82;!208,!#82]



# STYLISH SMARTS

- Like programming languages, the formal specification or grammar defines what is acceptable, but often provide no guidance on preferred forms.
- Good style improves readability.
  - SMILES: [13CH3-]
  - SMARTS1? [#6&13;H3-1A]
  - SMARTS2? [!c13-#6H3]
  - SMARTS3? [13&C;H3&-]
- Good style sometimes benefits performance
  - [C,c] -> [#6]      [X0D0R0h0A] -> [X0]    [#26] -> [Fe]



# ROBOCHEMISTRY AND THE BLACK SWAN

- A common source of “poor” SMARTS is observational bias, where an author makes assumptions about chemistry based on limited sampling.
- These SMARTS patterns are “good enough for government work”.
- An author may use the SMARTS “C” when they probably intended “[Cv4+0]”, after all aren’t all carbons neutral and four valent.
- Transformation rules misfiring on unanticipated input data leads to what PubChem calls “robochemistry”.



# POP QUIZ: FIND BENZENE

- Consider the challenge of finding benzene and only benzene in a SMARTS search.
- c1ccccc1 is a substructure.
- c:1:c:c:c:c:c:1 just ensures the bonds are aromatic.
- [0cv4+0]:1:[0cv4+0]:[0cv4+0]:[0cv4+0]:[0cv4+0]:...
- [0cD2h1+0]:1:[0cD2h1+0]:[0cD2h1+0]:[0cD2h1+0]:...
- [0cX3H1+0]:1:[0cX3H1+0]:[0cX3H1+0]:[0cX3H1+0]:...
- [0c+0;D2h1,D3h0\$(\*-[0#1D1h0+0])]:1:...



# POP QUIZ: MATCH NITRO GROUPS

- Likewise, if we assume RDKit has normalized nitro groups to \*[N+](=O)[O-]
- [NX3v4+0](=[OD1h0+0])[OD1h0-1]
- [ND3h0v4+0](=[OD1h0+0])[OD1h0-1]
- For carboxylic acids and carboxylates
- [CX3v4+0](=[OD1h0+0])[OD1;h0-1,h1+0]
- And esters
- [CX3v4+0](=[OD1h0+0])[O+0;D1h1,X2v2]



# RECURRING IDIOMS

- Degree, implicit hydrogen count and formal charge are often specified together:  $D^?h^?+^?$
- Connectivity and valence and formal charge are often specified together:  $X^?v^?+^?$



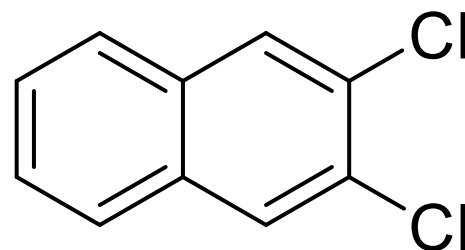
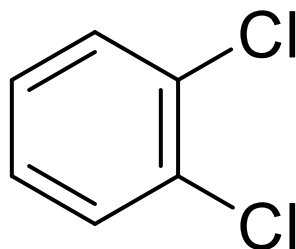
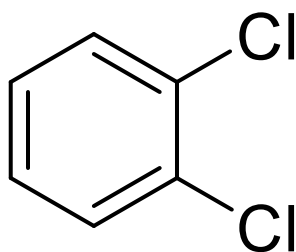
# SKETCH SEMANTICS

- SMARTS (mostly) has rigorous semantics that retrieve exactly what a cheminformatician asks for; unfortunately <1% of chemists can write SMARTS.
- The universal way of entering a substructure query is via a sketcher/structure editor such as JSME, Marvin, ChemDraw, Biovia Draw.
- Matching what a chemist believes they are asking for is a tricky challenge.



# THE ROOT OF ALL EVIL: AROMATICITY

- The classic challenge of chemical search is that 1,2-dichlorobenzene should match itself independent of the Kekule form used to represent it.



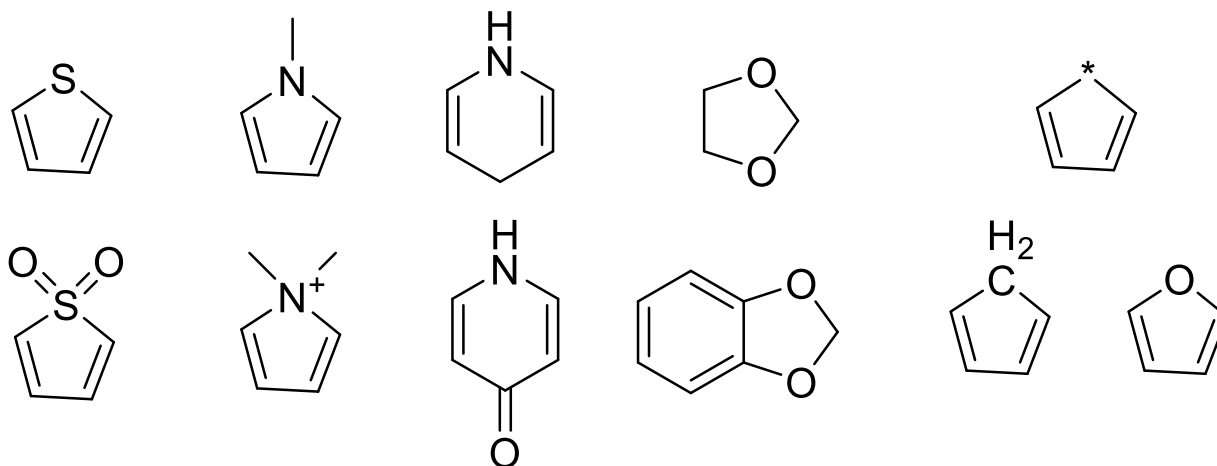
- In theory, chemists could mark bonds aromatic.
- Take home #1: Sketch bonds may be aromatic.





# THINGS ARE RARELY THAT SIMPLE

- These queries should(?) match the targets below.



- Queries drawn as aromatic may match aliphatic substructures, queries drawn as aliphatic may match aromatic substructures, and sometimes both.



# SKETCH SEMANTICS: 2<sup>ND</sup> ATTEMPT

- Allow atoms and bonds that could be aromatic, to be aromatic.
  - Benzene: [#6]1:,[#6][#6]:,[#6][#6]:,[#6]1
  - Cyclohexane: [#6]1[#6][#6][#6][#6][#6]1
  - Cubane: [#6]12[#6]3[#6]4[#6]1[#6]5[#6]4[#6]3[#6]25



# SKETCH SEMANTICS: 3<sup>RD</sup> ATTEMPT

- An atom that is unsaturated in the query, must be unsaturated in the target.
  - Benzene: [\*]1:=[\*][\*]:=[\*][\*]:=[\*]1
- Saturated atoms drawn with 2 neighbors may only be aromatic if saturated or unsaturated degree 3.
  - Cyclohexane: [\*;A,!i,D3]1[\*;A,!i,D3][\*;A,!i,D3][\*;A,!i,D3][\*;A,!i,D3][\*;A,!i,D3]1
- Saturated atoms drawn with 3 neighbors may only be aromatic if they remain saturated.
  - Cubane: [\*;A,!i]12[\*;A,!i]3[\*;A,!i]4[\*;A,!i]1[\*;A,!i]5[\*;A,!i]4[\*;A,!i]3[\*;A,!i]25



# FEATURES FOR POWER USERS

- Biovia query conversion should also support all of the expected features: atom lists, halogens, ring/chain topology, substitution count, bond logic, ring bond count, saturation/unsaturation etc.
- One cool feature supported by RDKit (and NextMove Software) is support for ChemAxon's "M MRV SMA" field, which allows arbitrary SMARTS to be associated with an atom in a Biovia query.



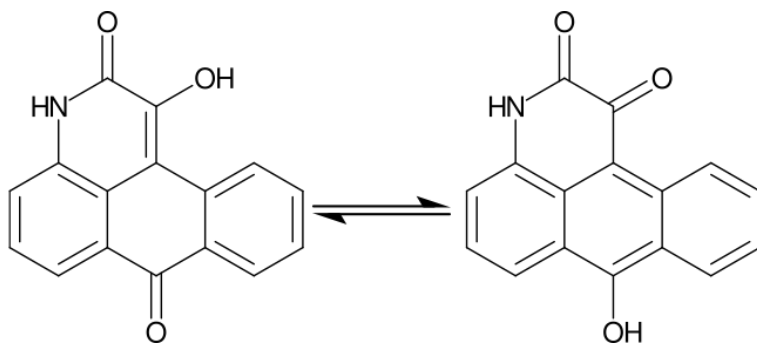
# SKETCH INTERPRETATION "SETTINGS"

- Lock Ring systems
  - A ring in a query always has to match a ring in the target. Locking ring systems prevents rings from forming parts of larger rings systems, i.e. benzene match naphthalene. Implemented by adding x2 constraints to ring atoms.
- Lock Chains
  - Acyclic query atoms must be acyclic in target.
- Lock Properties
  - Atoms without charge/isotope require no charge/isotope in target.
- Ignore stereochemistry/isotopes/charges.
  - Default behaviour of PubChem/CACTVS search.

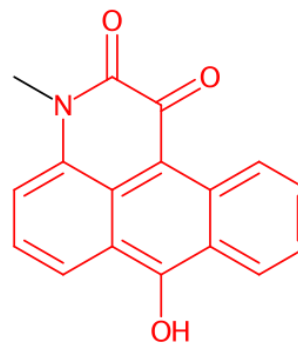


# THE PROBLEM WITH TAUTOMERS

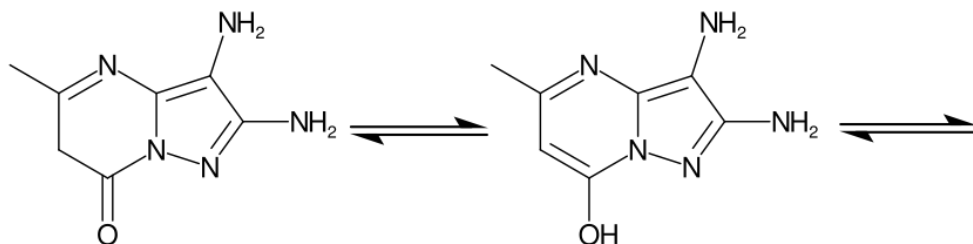
- Tautomerism causes problems for SMARTS matching.



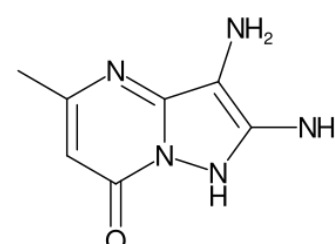
**quinone\_C filter**



**CID 5409668**



**het\_65\_G filter**

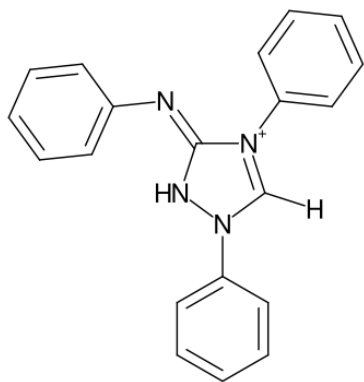


**CID 4060544**

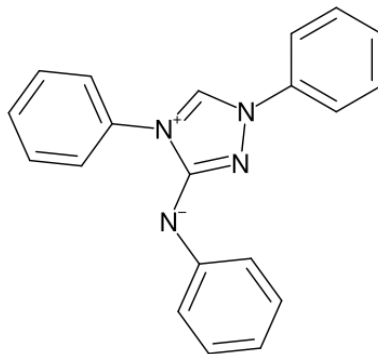


# AND WITH RESONANCE FORMS

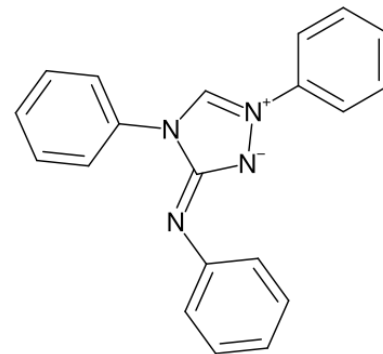
- As do resonance forms (and nitro representation &.)



het\_5\_inium filter



CID 720071  
(Oct 2018)



CID 720071  
(Feb 2019)



# ONE WAY OF SOLVING THIS

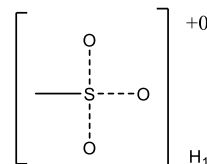
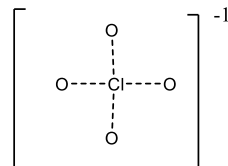
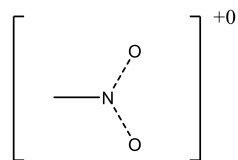
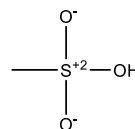
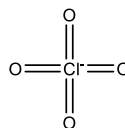
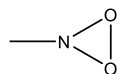
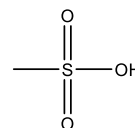
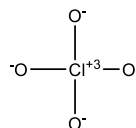
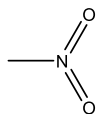
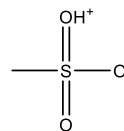
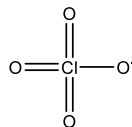
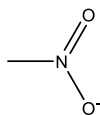
- Is to enumerate all possible (tautomeric/resonance) forms, and then search for each of them.
- This is the approach used by `RDKit::TautomerQuery` which uses `MolStandardize::TautomerEnumerator`.
- Something similar could in theory be implemented using `RDKit::ResonanceMolSupplier`.
- This extremely powerful technique can also solve the aromaticity problem (by enumerating Kekulé forms) and canonicalization (by enumerating SMILES paths).



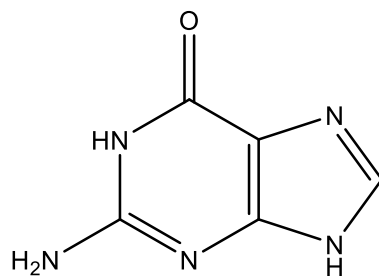


# DEMOCRITUS TO BORN-OPPENHEIMER

“All that exists are atoms in space all else is conjecture”



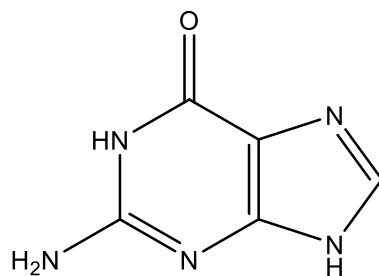
# GUANINE TAUTOMERS SMARTS



- Carbons are constrained to be sp<sup>2</sup>, heteroatoms to standard valences.
- (Non-triple) bonds become single, double or aromatic.
- [#7]1-=:[#6]-=:[#7]-=:[#6]2-=:[#6]-=:1-=:[#7]-=:[#6](-=:[#7]) -  
,=:[#7]-=:[#6] -=:2-=:[#8]
- Carbons: [#6X3v4i+0,#6X3v3-1,#6X3v3+1]
- Nitrogens: [#7X2v3+0,#7X3v3+0,#7X3v4+1,#7X4v4+1]
- Oxygens: [#8X1v2+0,#8X1v1-1,#8X2v2+0,#8X2v3+1,#8X3v3+1]



# GUANINE TAUTOMERS SMARTS

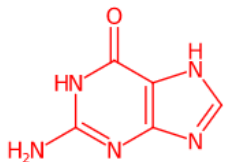
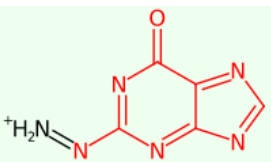
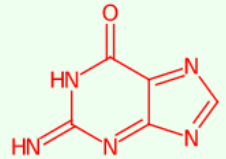
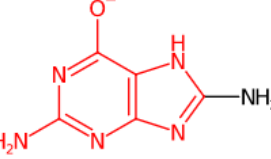
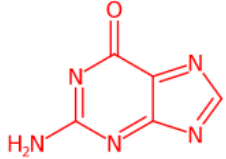
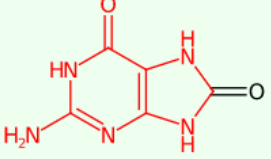
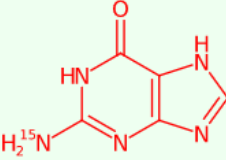
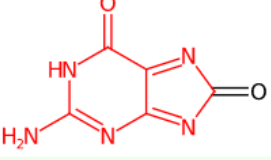
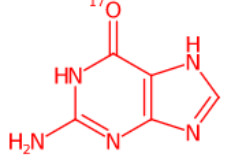
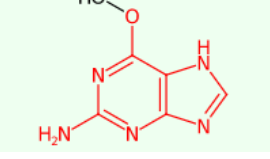


- [\*]1c2nc3c(ncn3C(=O)[nH]1)N1-,,: [\*]6c3c4c(ncn4C(=O)[nH]6)N-,=,:  
[\*]1c2nc3c(ncn3C(=O)[nH]1)N -,=, : [\*]6c3c4c(ncn4C(=O)[nH]6)N2-  
,,: [\*]6c3c4c(ncn4C(=O)[nH]6)N-,=, :1-,,: [\*]1c2nc3c(ncn3C(=O)[nH]1)N-,=, :  
[\*]6c3c4c(ncn4C(=O)[nH]6)N(-,=, : [\*]1c2nc3c(ncn3C(=O)[nH]1)N) -,=, :  
[\*]1c2nc3c(ncn3C(=O)[nH]1)N-,=, : [\*]6c3c4c(ncn4C(=O)[nH]6)N -,=, :2-,, :  
[\*]8c1c2c3c4c5c6c7c8c9c10c11c12c13c14c15c16c17c18c19c20c21c22c23c24c25c26c27c28c29c30c31c32c33c34c35c36c37c38c39c40c41c42c43c44c45c46c47c48c49c50c51c52c53c54c55c56c57c58c59c60c61c62c63c64c65c66c67c68c69c70c71c72c73c74c75c76c77c78c79c80c81c82c83c84c85c86c87c88c89c90c91c92c93c94c95c96c97c98c99c100c101c102c103c104c105c106c107c108c109c110c111c112c113c114c115c116c117c118c119c120c121c122c123c124c125c126c127c128c129c130c131c132c133c134c135c136c137c138c139c140c141c142c143c144c145c146c147c148c149c150c151c152c153c154c155c156c157c158c159c160c161c162c163c164c165c166c167c168c169c170c171c172c173c174c175c176c177c178c179c180c181c182c183c184c185c186c187c188c189c190c191c192c193c194c195c196c197c198c199c200c201c202c203c204c205c206c207c208c209c210c211c212c213c214c215c216c217c218c219c220c221c222c223c224c225c226c227c228c229c230c231c232c233c234c235c236c237c238c239c240c241c242c243c244c245c246c247c248c249c250c251c252c253c254c255c256c257c258c259c260c261c262c263c264c265c266c267c268c269c270c271c272c273c274c275c276c277c278c279c280c281c282c283c284c285c286c287c288c289c290c291c292c293c294c295c296c297c298c299c300c301c302c303c304c305c306c307c308c309c310c311c312c313c314c315c316c317c318c319c320c321c322c323c324c325c326c327c328c329c330c331c332c333c334c335c336c337c338c339c340c341c342c343c344c345c346c347c348c349c350c351c352c353c354c355c356c357c358c359c360c361c362c363c364c365c366c367c368c369c370c371c372c373c374c375c376c377c378c379c380c381c382c383c384c385c386c387c388c389c390c391c392c393c394c395c396c397c398c399c400c401c402c403c404c405c406c407c408c409c410c411c412c413c414c415c416c417c418c419c420c421c422c423c424c425c426c427c428c429c430c431c432c433c434c435c436c437c438c439c440c441c442c443c444c445c446c447c448c449c450c451c452c453c454c455c456c457c458c459c460c461c462c463c464c465c466c467c468c469c470c471c472c473c474c475c476c477c478c479c480c481c482c483c484c485c486c487c488c489c490c491c492c493c494c495c496c497c498c499c500c501c502c503c504c505c506c507c508c509c510c511c512c513c514c515c516c517c518c519c520c521c522c523c524c525c526c527c528c529c530c531c532c533c534c535c536c537c538c539c540c541c542c543c544c545c546c547c548c549c550c551c552c553c554c555c556c557c558c559c560c561c562c563c564c565c566c567c568c569c570c571c572c573c574c575c576c577c578c579c580c581c582c583c584c585c586c587c588c589c590c591c592c593c594c595c596c597c598c599c600c601c602c603c604c605c606c607c608c609c610c611c612c613c614c615c616c617c618c619c620c621c622c623c624c625c626c627c628c629c630c631c632c633c634c635c636c637c638c639c640c641c642c643c644c645c646c647c648c649c650c651c652c653c654c655c656c657c658c659c660c661c662c663c664c665c666c667c668c669c670c671c672c673c674c675c676c677c678c679c680c681c682c683c684c685c686c687c688c689c690c691c692c693c694c695c696c697c698c699c700c701c702c703c704c705c706c707c708c709c710c711c712c713c714c715c716c717c718c719c720c721c722c723c724c725c726c727c728c729c730c731c732c733c734c735c736c737c738c739c740c741c742c743c744c745c746c747c748c749c750c751c752c753c754c755c756c757c758c759c760c761c762c763c764c765c766c767c768c769c770c771c772c773c774c775c776c777c778c779c780c781c782c783c784c785c786c787c788c789c790c791c792c793c794c795c796c797c798c799c800c801c802c803c804c805c806c807c808c809c810c811c812c813c814c815c816c817c818c819c820c821c822c823c824c825c826c827c828c829c830c831c832c833c834c835c836c837c838c839c840c841c842c843c844c845c846c847c848c849c850c851c852c853c854c855c856c857c858c859c860c861c862c863c864c865c866c867c868c869c870c871c872c873c874c875c876c877c878c879c880c881c882c883c884c885c886c887c888c889c890c891c892c893c894c895c896c897c898c899c900c901c902c903c904c905c906c907c908c909c910c911c912c913c914c915c916c917c918c919c920c921c922c923c924c925c926c927c928c929c930c931c932c933c934c935c936c937c938c939c940c941c942c943c944c945c946c947c948c949c950c951c952c953c954c955c956c957c958c959c960c961c962c963c964c965c966c967c968c969c970c971c972c973c974c975c976c977c978c979c980c981c982c983c984c985c986c987c988c989c990c991c992c993c994c995c996c997c998c999



# GUANINE TAUTOMERS RESULTS

- Without post-processing, the raw results look like:

1		<b>135398634</b> C <sub>5</sub> H <sub>5</sub> N <sub>5</sub> O 151.126	54		<b>101930028</b> C <sub>5</sub> H <sub>5</sub> N <sub>6</sub> O 161.101
2		<b>140355140</b> C <sub>5</sub> H <sub>3</sub> N <sub>5</sub> O 149.110	55		<b>102341194</b> C <sub>5</sub> H <sub>5</sub> N <sub>6</sub> O 165.133
3		<b>504258</b> C <sub>5</sub> H <sub>3</sub> N <sub>5</sub> O 149.110	56		<b>135420630</b> C <sub>5</sub> H <sub>5</sub> N <sub>5</sub> O <sub>2</sub> 167.125
4		<b>135537618</b> C <sub>5</sub> H <sub>3</sub> N <sub>5</sub> O 150.104	57		<b>135498233</b> C <sub>5</sub> H <sub>3</sub> N <sub>5</sub> O <sub>2</sub> 165.109
5		<b>136689991</b> C <sub>5</sub> H <sub>5</sub> N <sub>5</sub> O 152.126	58		<b>141357953</b> C <sub>5</sub> H <sub>5</sub> N <sub>5</sub> O <sub>2</sub> 167.125



# NETQ & TOTX: CONSERVATION OF X

- Identifying/filtering relevant tautomers (perhaps via `SubstructMatchParameters::extraFinalCheck`)

```
int netq = 0;
unsigned int totx = 0;
for (unsigned int idx : match) {
    Atom *atm = mol->getAtomWithIdx(idx);
    netq += atm->getFormalCharge();
    totx += atm->getTotalDegree();
}
// Gaunine tautomers have totx of 29, and 0 netq.
return totx+netq >= 29;
```

- Under reasonable valence constraints  $\Sigma(X_i + q_i)$  is a conserved quantity of substituted tautomers/protomers.



# NEXT GEN. SMARTS OPTIMIZATION

- A significant development in the field of SMARTS optimization has been the development of next generation search engines.
  1. Uniform performance of all non-recursive atom expressions through the use of atom typing.
    - No penalty for [I,Br,F,Cl,C,c,#6,C]
  2. Atom and bond ordering in SMARTS patterns is now irrelevant.
    - BrCCC is no longer faster than CCCBr.
  3. Complex/distinctive expressions are now optimal.
    - [CD>3v4Z5Z6i0+0]



# (PATSY) SMARTS OPTIMIZATION 2020

- Intramolecular H Bonds
  - Input: O=[C,N]aa[N,O;!H0]
  - Internal: [O&i]=[C,N;D>1v>2i]-[aD3]:[aD3]-[N,O;!H0!D0]
- Biaryl atropisomers
  - Input: [aD3]a!@-a([aD3])[aD3]
  - Internal: [aD3]:[aD3x2]!@-[aD3x2](:[aD3]):[aD3]
- Cubane
  - Input: C12C3C4C1C5C4C3C25
  - Internal: [CZ4Z6Z8x>2]12-@[CZ4Z6Z8x>2]3-@[CZ4Z6Z8x>2]4-@[CZ4Z6Z8x>2]-@1-@[CZ4Z6Z8x>2]5-@[CZ4Z6Z8x>2]-@4-@[CZ4Z6Z8x>2]-@3-@[CZ4Z6Z8x>2]-@2-@5



# SUMMARY & CONCLUSIONS

- SMARTS is incredibly expressive.
- Writing SMARTS well is challenging.
- Divergence in implementations doesn't help.
- The situation is (hopefully) improving.
- Significant recent technical advances.
- Sketch semantics lowers the barrier to entry.
- But opens up new opportunities/challenges.





# ACKNOWLEDGEMENTS

- John Mayfield, NextMove Software
- Richard Gowers, NextMove Software
- Ingvar Lagerstedt, NextMove Software
  
- Greg Landrum
- David Weininger
- Wolf-Dietrich Ihlenfeldt
- OpenEye Scientific Software
- ChemAxon



# FOR GREG #1: FASTER ATOM MATCHING

- Current code

```
static inline int makeAtomType(int atomic_num, bool aromatic) {  
    return atomic_num + 1000 * static_cast<int>(aromatic);  
}  
  
static inline void parseAtomType(int val, int &atomic_num, bool &aromatic) {  
    if (val > 1000) {  
        aromatic = true;  
        atomic_num = val - 1000;  
    } else {  
        aromatic = false;  
        atomic_num = val;  
    }  
}
```

- Suggestion

```
static inline int makeAtomType(int atomic_num, bool aromatic) {  
    return (atomic_num<<1) + (aromatic?1:0);  
}  
  
static inline void parseAtomType(int val, int &atomic_num, bool &aromatic) {  
    atomic_num = val >> 1;  
    aromatic = val & 1;  
}
```



# FOR GREG #2: FASTER ATOM METHODS

```
namespace RDKit {  
  struct FastAtom : public Atom {  
    ROMol &fast_getOwningMol() const { return *dp_mol; }  
  };  
}  
  
RDKit::ROMol &Atom_getOwningMol_old(const RDKit::Atom *at) {  
  return at->getOwningMol();  
}  
  
RDKit::ROMol &Atom_getOwningMol_new(const RDKit::Atom *at) {  
  return ((const RDKit::FastAtom*)at)->fast_getOwningMol();  
}  
  
unsigned int Atom_getDegree_new(const RDKit::Atom *at) {  
  unsigned int idx = at->getIdx();  
  const RDKit::MolGraph &graph = at->getOwningMol().getTopology();  
  return (unsigned int)graph.m_vertices[idx].m_out_edges.size();  
}  
  
unsigned int Atom_getDegree_newer(const RDKit::Atom *at) {  
  unsigned int idx = at->getIdx();  
  const RDKit::FastAtom *fat = (const RDKit::FastAtom *)at;  
  const RDKit::MolGraph &graph = fat->fast_getOwningMol().getTopology();  
  return (unsigned int)graph.m_vertices[idx].m_out_edges.size();  
}
```

