Programová dokumentácia
Konverzia medzi vzorcami a názvami chemických zlúčenín
Radoslav Glinský - 2009

Zadanie:

V skratke program umožnuje konverziu medzi vzorcami a názvami chemických zlúčenín. To znamená, že k vstupnej chemickej zlúčenine program program rozpozná a vypíše jej chemický vzorec. Takisto k zadanému vzorcu program zistí jej slovný ekvivalent. Napríklad:
Vzorec: HCl
Názov: kyselina chlorovodikova

Program umožnuje konverziu medzi vzorcami a názvami týchto chemických zlúčenín:
Hydridy, oxidy, hydroxidy, halogenidy (soli halogénových kyselín), halogénové kyseliny, kyslíkaté kyseliny.

Ak to bolo možné, riešenie bolo implementované tak, aby odfiltrovalo aj neexistujúce zlúčeniny a nezmyselné názvy. Táto filtrácia funguje na základe zoznamu oxidačných čísel. Avšak toto riešenie občas odfiltruje aj zlúčeniny existujúce, ktorých existencia vyplýva z hlbších znalostí chémie, preto napríklad amónny katión nemožno použiť, pretože štvrtý vodík je viazaný vodíkovým mostíkom a analýza takýchto problémov by žiadala hlboké znalosti chémie, mnohonásobne by prevýšila rozsah zápočtového projektu a pre logické programovanie by bola skôr nevhodná. Program je implementovaný v jazyku Prolog, funkčnosť je garantovaná vo freeware implementácii SWI.

Užívateľské rozhranie:

Je veľmi jednoduché, používame len 2 predikáty:
Vzorec na meno: formula_to_name(+Vzorec,-Meno).
Meno na vzorec: name_to_formula(+Meno,-Vzorec).

Tvar vstupných dát (tvary vzorcov pre jednotlivé triedy zlúčenín):

Vzorec na meno : formula_to_name(+Vzorec,-Meno).

Vzorec zadávame bez medzier, prvé písmeno prvku je vždy veľké, druhé je vždy malé. Veľkosť čísel je maximálne dvojciferná. Vstup je uzavretý v znakoch ’.

Hydridy:

Tvar : **XHN**, kde X je prvok, H je vodík a N je prípadné číslo.

Oxidy:

Tvar : **XN1ON2**, kde X je prvok, N1 je prípadne jeho počet, O je kyslík, N2 jeho prípadná početnosť.

Hydroxidy:

Tvar : **XOH**, kde X je prvok a OH je hydroxidová skupina alebo **X(OH)N**, kde X je prvok, nasledujú zátvorky, OH je hydroxidová skupina, N je počet hydroxidových skupín.

Halogenidy:

Tvar : **XHN**, kde X je prvok, H je halogén, N je jeho početnosť.

Halogénové kyseliny:

Tvar : **HHal**, kde H je vodík a Hal je halogén.

Kyslíkaté kyseliny:

Tvar : **HN1EN2ON3**, kde H je vodík, N1 je prípadná jeho početnosť, E je kyselinotvorný prvok, N2 je jeho početnosť, O je kyslík a N3 je jeho prípadná početnosť.

Meno na vzorec : name_to_formula(+Meno,-Vzorec).

Chcel by som upozorniť, že diakritika v názvoch zlúčenín nie je povolená.

Hydridy:

Príklad: ‘hydrid sodny‘, ‘hydrid draselny’, ‘hydrid hlinity’

Hydroxidy:

Príklad: ‘hydroxid sodny’, ‘hydroxid vapenaty’, ‘hyroxid hlinity’

Oxidy:

Príklad: ‘oxid sodny’, ‘oxid mednaty’, ‘oxid hlinity’, ‘oxid kremicity’

Halogenidy:

Príklad: ‘chlorid sodny’, ‘chlorid barnaty’, ‘jodid hlinity’

Halogénové kyseliny:

Príklad: ‘kyselina chlorovodikova’, ‘kyselina bromovodikova’

Kyslíkaté kyseliny:

Príklad: ‘kyselina dusicna’, ‘kyselina trihydrogenfosforecna’, ‘kyselina chlorista’

Algoritmus:

Základný algoritmus je v oboch smeroch v princípe rovnaký a priamočiari. Na vstup sa zavolá procedúra schopná pomocou naprogramovaných regulárnych výrazov rozlíšiť, o aký typ zlúčeniny sa jedná. V prípade napasovania pustí príslušnu pomenovávaciu procedúru, ktorá už vzorec alebo názov rozoberie, rozanalyzuje a medzivýsledky zlepí do výstupu. Riešenie pomocou regulárnych výrazov som zvolil kvôli jednoznačnosti a ľahšiemu vylúčeniu zlých vstupov, čo je vyvážené vyššou časovou zložitosťou, ktorá sa ale pre stanovený rozsah dá zanedbať.

Program – 1. časť: zo vzorca na názov:

Po volaní procedury **formula_to_name(+Formula,-Name)** – pusti procedúru **analyze_formula(+Fromula,-Name)**, ktorá postupne pre každú triedu zlúčeniny spustí procedúru **reg_match(+RegExp,+String).** Príklad volania môže byť nasledujúci:

```
analyze_formula(Formula,Name):- reg_match('[A..Z]{0,1}[a..z]H$',Formula),
name_hydride(Formula,Name), !.
```

Procedúra reg_match skúsi namatchovať zadaný regulárny výraz na vzorec a ak uspeje, pustí príslušnú procedúru **name_zlucenina(+Formula,-Name)**. Časť „**zlucenina“** reprezentuje slová „**hydride“**, „**oxide“**, „**hydroxide“**, „**halogen_acid“**, **„halogenid“**, „**ox_acid“**, ktoré reprezentujú funkcie na prevod hydridov, oxidov, hydroxidov, halogenových kyselín, halogenidov a kyslíkatých kyselín.

V spomínanom príklade sa konkrétne snaží predikát namatchovať zadaný vzorec na vzorec hydridu.

Ako to aj vidíme na použitom regulárnom výraze:

Najprv musí byť jedno veľké písmeno.

Potom žiadne alebo jedno malé písmeno (ak je značka prvku dvojpísmenná).

A nakoniec jedno písmeno 'H' označujúce vodík.

Ak sme teda už našli správny typ zlúčeniny, postupujeme dalej v precedúre name_zlucenina.
Hydridy:
Zjednodušené volanie vyzerá takto:

```
name_hydride(Formula,Name):- hydride_cation(Formula,Cation,Rest),
                             hydride_anion(Rest,Multi), num_suffix(Multi,Suffix),
                             element(_,Base,Cation,OxList), member(Multi,OxList),
                             concatenate(['hydrid',' ',Base,Suffix,'y'],Name).
```

hydride_cation - zistí značku katiónu a vráti zbytok zoznamu
hydride_anion – zo zbytku vráti počet atómov vodíka
num_suffix – podľa počtu atómov vodíka určí koncovku katiónu,
napríklad: num_suffix(2,'nat').
element – podľa značky katiónu zistí jeho slovný základ a rôzne oxidačné čísla, ktoré môže nadobúdať,
napríklad: element('sodik','sod','Na',[1]).
member – kontrola, či zadaný počet atómov vodíka splna oxidačné možnosti katiónu

Takže zo vzorca sme postupne slovný základ katiónu, jeho koncovku a teda precedúra
concatenate(['hydrid',' ',Base,Suffix,'y'],Name)
nám jednotlivé stringy pospája do jedného výsledného Name.

Oxidy, hydroxidy, halogenidy:
Pri oxidoch a hydridoch postupujeme veľmi podobne, ako pri hydridoch. Zásadnejší rozdiel je len v tom, že musíme zo zadaného stringu „vysekávať“ aj početnosť jednotlivých atómov.

Halogénové kyseliny:
Sú riešené jednoducho, stačí vystrihnúť meno kyselinotvorného prvku. Potom už len pospájame stringy:
concatenate(['kyselina',' ',Base,'o','vodikova'],Name).
V tomto prípade je Base slovný základ kyselinotvorného prvku.

Kyslíkaté kyseliny
V procedúrach:
acid_hydrogen_num – oddelíme vodíky a ich počty
acid_element – vystrihneme kyselinotvorný prvok
acid_element_num – jeho prípadný počet
acid_oxygen_num - a zistíme počet atómov kyslíka

Následne sa tieto čísla dosadia do vzorca a tak získame koncovku.
Neurčitosť koncovky -ičný, -ečný rieši procedúra **suffix_exception**, ktorá má vymenované prvky s druhou koncovkou, napríklad: suffix_exception('P',5,'ecn'), takže bude fosfor-ečný.

Program – 2. časť: z názvu na vzorec:

Procedúra **name_to_formula(+Meno,-Vzorec)** spustí procedúru **analyze_name(+Meno,-Vzorec)**. Procedúra následne spúšťa osobitne pre dané typy zlúčenín **reg_match(+RegExp,+Meno)** a skúsi napasovať meno na nejaký vzor. Príklad volania, kedy sa snažíme namatchovať nejaký hydrid:
analyze_name(Name,Formula):- reg_match('hydrid *',Name), formulize_hydride(Name,Formula), !.

Hydridy - sú určené podľa začiatočných slov „hydrid“.
Oxidy - sú určené podľa začiatočných slov „hydrid“.
Hydroxidy sú určené podľa začiatočných slov „hydroxid“.
Halogenové kyseliny sú určené podľa prvého slova „kyselina“, medzera, potom je povolená postupnosť ľubovoľných znakov, koniec mena ale musi obsahovať reťazec „vodíková“.
Halogenidy sú určené koncovskou „-id“.
Kyslíkatú kyselinu možno rozoznať podľa prvého slova „kyselina“, medzery, série ľubovoľných znakov a posledného písmena 'a'.

Po namatchovaní sa následne pustí procedúra **formulize_zlucenina**, kde slovo „**zlucenina**“ reprezentuje slová „**hydride**“, „**oxide**“, „**hydroxide**“, „**halogenide**“, „**halogen_acid**“, „**ox_acid**“, prepisujúce v tomto poradí meno zlúčeniny na hydrid, oxid, hydroxid, halogenid, halogénovú kyselinu, kyslíkatú kyselinu.

Hydridy:
formulize_hydride(Name,Formula):- cut_suffix('y',Name,Rest),
cut_prefix('hydrid ',Rest,Rest2), cut_ox_suffix(Rest2,_,Num,Base),
element(_,Base,Sign,OxList), member(Num,OxList),
number_one(Num,OutNum), concatenate([Sign,'H',OutNum],Formula).

cut_suffix – odsekne príponu 'y'
cut_prefix – odsekne predponu 'hydrid'
cut_ox_suffix – k zbytku stringu, ktorý nám ostane, vráti oxidačné číslo a slovný základ atómu.
Toto číslo a základ získa tak, že postupne zo zbytku stringu sa pokúsi odrezať príponu. Ak sa to podarí, tak vráti príslušné oxidačné číslo. Slovný základ vznikne po odrezaní prípony.
element – prevedie slovný základ na značku
member – kontrola správnosti oxidačných čísel
number_one - ak je vstup jednička, vráti prázdny reťazec, ak je väčší, vráti vstup.
Slúži na to, aby sme početnosť atómu 1 vôbec nepísali.
concatenate – vytvorí finálny vzorec spojením značky prvku, vodíka a početnosti vodíka

Oxidy:
Pri oxidoch je situácia iná len v hľadaní počtov kyslíku. Pomer zistíme ľahko z koncovky oxidu a znalosti, že kyslík má v oxidoch oxidačné číslo – II. Avšak toto číslo nemusí byť celé, preto je potrebné zdvojnásobiť oxidačné číslo katiónu a rekurzívne na neho pustiť znovu tú istú procedúru. To nám zabezpečí procedúra oxide_numbers – vráti počty atómov prvku a kyslíka

Hydroxidy:
Postup je veľmi podobný ako pri hydridoch, akurát ošetrujeme prípad, kedy je potrebné ozátvorkovať hydroxidovú skupinu OH.

Halogénové kyseliny:
Prevod je triviálný. Odsekneme predponu „kyselina“ a príponu „ovodikova“ a zo zbytku určíme kation.

Halogenidy:
Takisto prevod podobný ako pri hydridoch. Rozdiel je len v nájdení aniónu v prvom slove, kde odstránením koncovky „-id“ dostaneme slovný základ halogénu.

Kyslíkaté kyseliny:
Pri analýe kyslíkatých kyselín vysekneme prvé slovo „kyselina“ a venujeme sa zbytku, z ktorého ešte odstránime posledné písmeno 'a', aby sme sa dostali k jadru koncovky určujúcej oxidačné číslo kyselinotvorného prvku. Ak už máme získané toto oxidačné číslo, zistíme si počty kyslíkov a vodíkov tak, aby celková zlúčenina bola neutrálna. Slúži k tomu procedúra **find_hydro_ox(+OxEl,+HNum,+ONum,-OutHNum,-OutONum)**. Jej argumenty sú oxidačné číslo kyselinotvorného prvku, vstupný počet vodíkov a kyslíkov a výstup je počet vodíkov a kyslíkov. **HNum** a **ONum** inicializujeme číslom jedna. Skúsime dosadiť do vzorca tak, aby suma oxidačných čísel bola nulová. Ak je väčšia ako nula, potom je treba zvýšiť počet kyslíkov, opačne zase počet vodíkov. Počty sú obmedzené aby nedošlo k pretečeniu na 10 kyslíkov a 20 vodíkov. Teraz už máme všetky potrebné informácie na zostavenie vzorca a následne voláme:
concatenate(['H',HNum2,Sign,'O',ONum2],Formula),
kde Hnum2 a Onum2 sú počty vodíkov a kyslíkov, Sign je značka kyselinotvorného prvku.

Možnosti:

Do progamu by sa dali doplniť soli kyslíkatých zlúčenín, komplexné zlúčeniny, podvojné soli, doplniť prevod z mena na vzorec o polykyseliny. U ostatných zlúčenín by to nemuselo byť vhodné kvoli spôsobu určovania existencie z oxidačných čísel. Tento spôsob však môže odfiltrovať aj niektoré existujúce zlúčeniny kvôli absurdným podmienkam ich vzniku ako sú napríklad oxidy vzácnych plynov.