Kapitola 25

Státnice I3: Generování přirozeného jazyka

25.1 Úvod

Problémy NLG (Natural Language Generation): Jak by měly počítače komunikovat s člověkem? Jaké chování od nich lidi očekávají? (Někdy je lepší vyplivnout graf nebo tabulku.) Co je "srozumitelný" jazyk v dané situaci? Jak převést reprezentaci znalostí (často hromada numerických dat) do "lidské podoby" (typicky malý počet abstraktních pojmů)?

Zahrnuje AI, lingvistické formální modely. Vlastně inverzní k analýze, k porozumění – nejde tu ale o zabývání se hypotézami, ale výběr vhodné strategie sdělení. "Dvojsměrný" systém se staví dost těžko – analýza musí počítat s nekorektním vstupem, ale neřeší srozumitelnost svého výstupu. Reprezentace znalostí v obou typech systémů je většinou odlišná.

Aplikace: většinou prezentace informací, které vznikají automaticky, případně (částečná) automatizace rutinní dokumentace (lékařské, programátorské atd.). Důležité, protože interní reprezentace databází nejsou člověku srozumitelné.

Důvody použití:

- Konzistence textů a dat
- Splnění standardů pro formát výstupu
- Rychlost produkce dokumentů
- Mnohojazyčnost
- Lidi to prostě nudí, je-li to monotónní úkon.

Historie

Od 50-60. let, v rámci MT systémů, první formální gramatiky pro náhodné generování korektních vět. V 70. letech první pokusy o NLG pro interpretaci dat. Skutečné nasazení systémů v 90. letech.

25.2 Struktura NLG systému

Není úplně ustálená, je mnoho možností.

- Vstup: zdroj znalostí, komunikativní cíl (konkrétního použití – např. "shrnout data o počasí za poslední měsíc"), uživatelský model ("charakterizace cílového publika"), historie diskurzu ("co už bylo řečeno")
- Výstup: text (formátování záleží na aplikaci — často např. HTML)

Typická základní architektura – pipeline:

- **Plánovač dokumentu (plánovač textu)** (určí obsah a strukturu výstupu)
  - Vytváří obsah ("zprávy"), určuje, které z nich je třeba vypsat pro splnění komunikativního cíle (content determination – výběr relevantních informací)
  - Strukturuje výstupní dokument, aby bylo možné vygenerovat srozumitelný a souvislý text (document structuring) – podle očekávání čtenáře (žánr), seskupování související informace (např. "vše o teplotě napsat za sebou")
  - Rozdělení do vět a jazykové otázky se tady zatím neřeší
- **Mikroplánování (plánovač vět)** (jaká slova, syntaktické struktury atp. použít)
  - Někdy je výstupem už text, někdy mezireprezentace (např. specifikace času věty apod.)

  - Lexikalizace (jaká slova a konstrukce použít – "pršelo od 11. do 14. / pršelo 11.,12.,13.,14."?), generování označení (refering expression generation – jak označovat entity? – první / následné zmínky)
  - Agregace (mapování struktury dokumentu na jazykovou strukturu – co vecpat do které věty/odstavce?) – např. "Minulý měsíc byl chladný" + "Minulý měsíc byl suchý" = "Minulý měsíc byl chladný a suchý"

- **Povrchová (lingvistická) realizace** (převod abstraktní reprezentace použité mikroplánovačem do skutečného textu)
  - Realizace lingvistická i strukturální (formátování textu)
  - Některé systémy (PEBA) mají dost jednoduchou jazykovou realizaci – jde vlastně o šablony doplňované údaji
  - Jiné (ModelExplainer) používají abstraktní synt. struktury, hodí se i pro mnohojazyčný výstup
  - Systemic Grammar, Functional Unification Grammar

Datové mezistupně:

- Plán dokumentu – typicky stromová struktura, vnitřní uzly – sturkturní informace, listy – obsah ("zprávy")
- Specifikace textu – opět stromy, vnitřní uzly – sturktura textu, listy – věty ("specifikace frází")
  - Specifikace fráze: ortografický řetězec (vše vyřešené) / canned text (nutné řešit velká písmena, interpunkci apod.), abstraktní syntaktická sturktura (lexémy a jejich rysy, závislostně uspořádané), lexicalized case frame (spíše lexikalizovaný sémantický strom).

25.3 Příklady NLG systémů

- KPML – obecný NLG systém, Systemic Functional Grammar (pro několik jazyků, vč. EN, CZ)
  - SFG – výběr z alternativ: povrchová realizace je důsledkem výběru funkčních rysů (z popisu/taxonomie celého jazyka – systemic network), znamená projití sítí (krok = výběr rysu) bez backtrackingu (každý přechod má určité podmínky, napojení = závislost jazyk. elementů)

Počítač jako pomůcka

- FoG – generování předpovědi počasí (v Kanadě – EN/FR)
- PlanDoc – dokumentace navrhovaných změn v telefonních sítích
- AlethGen – pomůcka při psaní odpovědí zákazníkům :-)
- Drafter – pomůcka pro psaní manuálů k softwaru

Počítač jako samostatný autor

- IDAS – poskytuje informace o používání přístroje na základě reprezentace znalostí
- ModelExplainer – vysvětluje strukturu objektově-orientovaných programů
- PEBA – popisy taxonomické báze znalostí
- Piglet – vysvětluje pacientům v nemocnici jejich lékařské zprávy
- STOP – generuje personalizovanou informaci o škodlivosti kouření :-)

25.4 Evaluace

- Založená na úkolech (jak systém pomáhá člověku zvládnout daný úkol)
- Lidská (lidi posuzují srozumitelnost)
- BLEU nebo něco podobného