言語解析論
レポート

講師　竹内孔一

提出日と形式

- 提出物(これで評価の40%)
  - レポート
- 提出日
  - 7月27日(金)
- 形式
  - なるべくワープロで出力したものを提出
  - 他人のコピーで作成しないように
  - Moodleで pdfか wordで提出
    - 紙で出す人は要相談

課題

1. 形態素解析器MeCabを演習室のlinuxにいれてインストールする
   - rootにならず，ローカルにインストール
   - インストール手順を説明する
   - 無理ならばwindows版など自分のマシンでも良い
   - Linuxに入れた人の方を少し良く評価します
2. 新単語を登録して解析する
   - 検索で「mecab　単語　追加」など
   - 辞書に無い単語を追加して，出力されることを確認する

3. 形態素解析器MeCabの出力についていくつかの例文を入れてみて評価する
   - 評価とは数字的なものも考えられるが，典型的な例を入れて考察する
   - 極端な例(英語を入力)などしないこと(無意味)
   - これから使う人にどの程度信用できそうなものかを実例をもって説明する

評価のポイント

- レポートの内容
  - 感想文になっていないか?
  - Linuxでのインストール手順(簡単で良い)(情報系のみ)
  - 形態素解析がどういう点でうまくできていて，どういう点では問題があるか，事例とともに説明されているか

MeCabインストール

- MeCab本体
  - http://taku910.github.io/mecab/
  - からmecab-0.996.tar.gzをdownload
  - Blogなど参考にインストール方法を探すこと
- 辞書
  - IPA辞書をdownload
- 文字コード
  - UTF-8 (IPA辞書はEUCなので変換が必要)
  - 演習室のターミナルはUTF-8なのでそのままいれると文字化けして出力が見られない

MeCabインストール

- tar.gzは

  $tar xvfz mecab-0.996.tar.gz

  $tar xvfz mecab-ipadic-2.7.0-20070801.tar.gz

  –とすると展開できる

- 手順

  –mecabインストール先をmkdir

  –mecab本体，辞書の順

  例) /home/koichi/study の下に入れたい場合

  $ mkdir /home/koichi/study

  $ cd mecab-0.996

$./configure --prefix /home/koichi/study --with-charset=utf8

–インストール先のパス，それから入出力と辞書を全てutf8に変更する

–これで，各linuxシステムに合わせたheaderなどができあがる.

–このconfigureをやり直したい場合，パスや文字コードを変えたい場合は，またファイルをけして.tar.gzファイルを展開するところからやり直す必要がある

- ディレクトリを消すとき rm –rf mecab-0.996

$make

–これでコンパイル

$make check

–double arrayなどのプログラムのテストが走る

$make install

–これで /home/koichi/study/の下にいろいろ必要なプログラムが入る

- 辞書を入れる

$cd ..

$cd mecab-ipadic-2.7.0-20070801

$./configure --prefix=/home/koichi/study --with-charset=utf8 --with-mecab-config=/home/koichi/study/bin/mecab-config

–ここの --with-mecab-configがとても大事．先ほど入れたプログラムを読み込むため

checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking whether make sets $(MAKE)... yes
checking for working aclocal-1.4... missing

– いろいろmissingメッセージがでるが動いている.

$make

$make install

- 実行

$cd /home/koichi/study/bin

$./mecab

(STDINで入力)リターンで解析結果が得られる

MeCab出力例

```
[koichi@grade bin]$ pwd
/home/koichi/study/bin
[koichi@grade bin]$ ./mecab
今日は良い天気ですね.
今日	名詞,副詞可能,*,*,*,*,今日,キョウ,キョー
は	助詞,係助詞,*,*,*,*,は,ハ,ワ
良い	形容詞,自立,*,*,形容詞・アウオ段,基本形,良い,ヨイ,ヨイ
天気	名詞,一般,*,*,*,*,天気,テンキ,テンキ
です	助動詞,*,*,*,特殊・デス,基本形,です,デス,デス
ね	助詞,終助詞,*,*,*,*,ね,ネ,ネ
.	記号,句点,*,*,*,*,.,.,.
EOS
```

止めるときは ctrl-c

MeCabの実行

- fileに日本語を書いておいて出力もできる

  ./mecab file > outfile

レポートの書き方

- 書くべき内容
  - どういう結論か書きましょう
  - 事例をつけて具体的に分析する(複数)
  - 出典を書く
  - 名前，出席番号を忘れずに書く
- 注意
  - 意味のある分析を書きましょう
  - 口語表現，冗長表現は避けましょう
  - 事例無しで憶測で書かない
  - ワープロを使いましょう

以降は課題遂行のための
参考のスライド

shared tool の利用

- 意義
  - インターネットを利用して自らの技術を提供(世界規模)
  - ネット上から必要な tool を利用して短時間で処理モデルを作成
- shared tool の成功例
  - オペレーティングシステムOS
    - 無料のOS linux システム (Redhat, vine, freeBSD)
    - 最初は個人が Unix を真似て作成した
- shared tool とは
  - C言語でのライブラリイ（例) html parser（Web解析)
  - 正規表現ライブラリイ，　perl の　CPAN
  - 研究上で開発されたソフト　形態素解析，係り受け解析
  - 数学モデル SVMs, HMM

>>得なことが多い

shared tool の利用

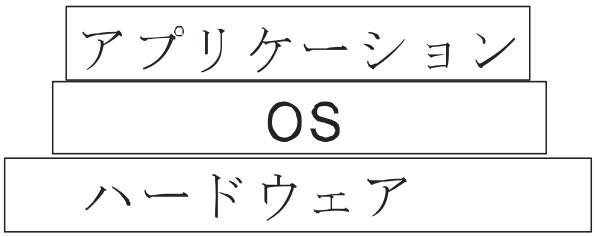
- 難しい
  - インストールが容易ではない
    - tool自身も共通ライブラリを使っている
- 注意点
  - 著作物としての使用権限の確認
    - 明記されているので必ず確認 (英語が多い)
    - どういう権限で利用，改良できるのか
    - 発表は研究のみか商用まで使えるのか?
    - free soft とは一般に 金額は0でも権利は放棄していない
  - セキュリティー
    - 共通するライブラリーまたはshared tool にセキュリティフォールがあると影響をうける
      →貴重な個人データの流出 (key loging)

インストール

- 必要な知識
  - インストール先のコンピュータの構成
  - インストールとは何か
  - インストール先との調整をとるシステム
  - インストールのよくある手順とエラーの理解
- 事例
  - 情報工学科計算機システムでC言語で書かれたプログラムをインストールする

コンピュータの構成(単体)

- 問題点
  - ハードウェア，ソフトウェアに多様性
    → インストールのやり方が異なる　　違いを認識する!
- 共通構成
  - ハードウェア>OS>各ソフト群
- 各OSの種類
  - ハードOS一体型
    - Mac OS X (Linux系)
    - 他のUnix システム　Irix, Ultrix
    - 携帯電話，Palm OS
  - 分離型　(IBM互換機)
    - MS WinXP, WinNT, Win2000, Win98
    - Linux　Redhat, vine, free-BSD, solaris



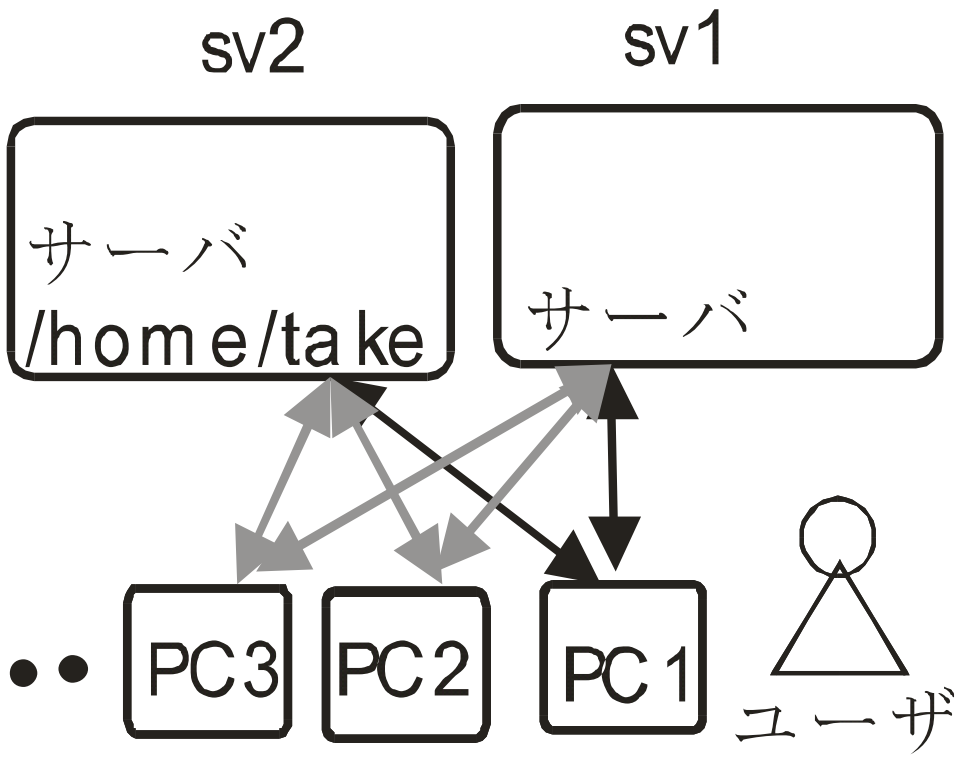
コンピュータの構成(複数)

- コンピュータシステム
  - 複数のPCが存在し管理されている
    - 会社，大学，研究室，計算機実習室，などあらゆる組織
      Windows, Linux, Max OS X.
  - ユーザによる制約
    - ファイルに対するアクセス権利 (管理者 or 一般ユーザ)
      書き込み，読み込み
  - ファイルとユーザの一元管理
    - どの席に座っても同じデータを見ることができる

  → 個人として書き込める領域が制限

一般的な構成

- 複数のPCによるシステム（例）



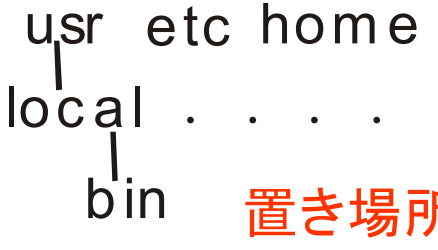
制約
ユーザの許可
sv2 の /home/take
PC1, sv1, sv2 はすべて
管理者パスが違う

利点
ユーザはどのPCでも
/home/take 以下なら同じ
ファイルを見ることができる
同じユーザ名でパスワードも
同じ

インストール

- プログラムをインストールとは
  - OSが管理する下に適切な機械語としてプログラムを置く
- ポイント
  - OSの管理下に的確に置く
    - ファイルの構造理解
  - 機械語(binary)の生成

例)linux



置き場所に決まり

例) C言語　コンパイラを通してbinary

共通ライブラリの利用 (例　stdio.h time.h)
→C言語でソースを書いてコンパイルすること
も一種のインストール

インストールの難しさ1

- 問題点
  - OSごとにライブラリの位置や使える関数が違う!!

  a. 個別OSにsource プログラムを書き直す

  b. OSごとにbinary (コンパイル済みのプログラム)を用意
    - 例) Windows 関連のソフト　windows のウィルス

  例) C言語のlibrary　#include time.h -> sys/time.h
- aの方法での解決策
  - 自動でOSの種類と使える library を調べてsource を書き換えてコンパイルする (Linux と Unix)

    configure 実行ファイル

インストールの難しさ2

- インストールと権限
  - 通常管理者権限にファイルを置く
    - root もしくは administrator 権限が必要
  - 複数台のPC環境(演習室，会社)
    ->管理者権限は与えられない
  - > どうするか?
- 解決策
  - インストール先を自分のユーザ権限の範囲に指定してインストールする
    - 可能(Linux, Unix)　不可 (Windows, Mac)

例 Win /I386/system32
Linux　/usr/local/bin/

インストールの概念的手順

- Linux の場合について
  - プログラムをdownload する
  - ./configure --prefix[インストール先]を実行

    OS, library をcheck して Makefileを作成
  - make を実行

    source プログラムがコンパイルされてbinary になる
  - make install

    binary がインストール先 にコピーされる

まとめ

- メディア処理として
  - 言語処理の必要性
  - 無料のソフトウェアによる処理モデルの可能性
  - ソフトウェアを install する上での必要な概念の提示