615.1.8 - Dynamic Programming

This is an R Markdown Notebook. You may be viewing this notebook in a web browser as a form of static HTML file, or you may be interactively working with document inside an RStudio. You may walk through next chunk of the code interactively using *Cmd+Option(alt)+Backtick(‘)* (in Mac OS X) or *Ctrl+Alt+N* (in Windows). You may also place your cursor inside a chunk and press *Cmd+Shift+Enter* (both in Mac OS X and in Windows), or press the small green button in the right-top-corner of each chunk.

1. Manhattan Tourist Problem - Cost only

```
#include <Rcpp.h>
#include <algorithm>

using namespace Rcpp;
using namespace std;

// We assume that C contains all negative values initially
double mtp_dp(NumericMatrix& H, NumericMatrix& V, int r, int c, NumericMatrix& C) {
  if ( C(r,c) < 0 ) { // need to compute and store
    if ( r == 0 ) {
      if ( c == 0 ) C(r,c) = 0;
      else C(r,c) = mtp_dp(H,V,r,c-1,C) + H(r,c-1);
    }
    else {
      if ( c == 0 ) C(r,c) = mtp_dp(H,V,r-1,c,C) + V(r-1,c);
      else {
        double hcost = mtp_dp(H,V,r,c-1,C) + H(r,c-1);
        double vcost = mtp_dp(H,V,r-1,c,C) + V(r-1,c);
        C(r,c) = hcost > vcost ? vcost : hcost;
      }
    }
  }
  return C(r,c);
}

// [[Rcpp::export]]
double mtp(NumericMatrix H, NumericMatrix V) {
  int r = H.nrow();
  int c = H.ncol() + 1;
  if ( ( V.nrow() + 1 != r ) || ( V.ncol() != c ) )
    stop("The dimensions of H and V do not match");
  NumericMatrix C(r, c);
  fill(C.begin(), C.end(), -1.0);

  return mtp_dp(H, V, r-1, c-1, C);
}
```

```
hcost <- matrix(c(4,7,6,1,1,2,4,8,6,5,0,5,1,4,8,7,9,0,7,5),5,4)
vcost <- matrix(c(0,9,1,3,6,7,8,6,6,1,4,6,2,0,8,0,4,6,9,7),4,5)
print(hcost)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    4    2    0    7
## [2,]    7    4    5    9
## [3,]    6    8    1    0
## [4,]    1    6    4    7
## [5,]    1    5    8    5
```

```
print(vcost)
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    0    6    6    2    4
## [2,]    9    7    1    0    6
## [3,]    1    8    4    8    9
## [4,]    3    6    6    0    7
```

```
mtp(hcost,vcost)
```

```
## [1] 21
```

2. Manhattan Tourist Problem with backtracking

```
#include <Rcpp.h>
#include <algorithm>
#include <string>

using namespace Rcpp;
using namespace std;

// We assume that C contains all negative values initially
double mtp2_dp(NumericMatrix& H, NumericMatrix& V, int r, int c, NumericMatrix& C, LogicalMatrix& P) {
  if ( C(r,c) < 0 ) { // need to compute and store
    if ( r == 0 ) {
      if ( c == 0 ) C(r,c) = 0;
      else {
        C(r,c) = mtp2_dp(H,V,r,c-1,C,P) + H(r,c-1);
        P(r,c) = true;
      }
    }
    else {
      if ( c == 0 ) {
        C(r,c) = mtp2_dp(H,V,r-1,c,C,P) + V(r-1,c);
        P(r,c) = false;
      }
      else {
        double hcost = mtp2_dp(H,V,r,c-1,C,P) + H(r,c-1);
        double vcost = mtp2_dp(H,V,r-1,c,C,P) + V(r-1,c);
        C(r,c) = hcost > vcost ? vcost : hcost;
        P(r,c) = hcost < vcost;
      }
    }
  }
  return C(r,c);
}
```

```
string mtp2_optpath(int32_t r, int32_t c, NumericMatrix& H, NumericMatrix& V, LogicalMatrix& P) {
  std::string path = "(" + to_string(r) + "," + to_string(c) + ")";
  while ( ( r > 0 ) || ( c > 0 ) ) {
    int w;
    if ( P(r,c) ) { w = H(r,c-1); --c; }
    else { w = V(r-1,c); --r; }
    path = string("(") + to_string(r) + "," + to_string(c) + ")--[" + to_string(w) + "]-->" + path;
  }
  return path;
}

// [[Rcpp::export]]
List mtp2(NumericMatrix H, NumericMatrix V) {
  int r = H.nrow();
  int c = H.ncol() + 1;
  if ( ( V.nrow() + 1 != r ) || ( V.ncol() != c ) )
    stop("The dimensions of H and V do not match");
  NumericMatrix C(r, c);
  LogicalMatrix P(r, c);
  fill(C.begin(), C.end(), -1.0);

  double optcost = mtp2_dp(H, V, r-1, c-1, C, P);
  string optpath = mtp2_optpath(r-1, c-1, H, V, P);
  return List::create(Named("cost")=optcost,
              Named("path")=optpath);
}
```

```
hcost <- matrix(c(4,7,6,1,1,2,4,8,6,5,0,5,1,4,8,7,9,0,7,5),5,4)
vcost <- matrix(c(0,9,1,3,6,7,8,6,6,1,4,6,2,0,8,0,4,6,9,7),4,5)
mtp2(hcost,vcost)
```

```
## $cost
## [1] 21
##
## $path
## [1] "(0,0)--[4]-->(0,1)--[2]-->(0,2)--[0]-->(0,3)--[2]-->(1,3)--[0]-->(2,3)--[8]-->(3,3)--[0]-->(4,3
```

3. Edit distance - cost only

```
#include <Rcpp.h>
#include <string>

using namespace Rcpp;
using namespace std;

int edist_dp(string& s1, string& s2, IntegerMatrix& cost, int r, int c) {
  if ( cost(r,c) < 0 ) {
    if ( r == 0 ) {
      if ( c == 0 ) cost(r,c) = 0;
      else cost(r,c) = edist_dp(s1, s2, cost, r, c-1) + 1;
    }
    else if ( c == 0 ) cost(r,c) = edist_dp(s1, s2, cost, r-1, c) + 1;
    else {
      int iDist = edist_dp(s1, s2, cost, r, c-1) + 1;
      int dDist = edist_dp(s1, s2, cost, r-1, c) + 1;
      int mDist = edist_dp(s1,s2,cost,r-1,c-1)+(s1[r-1]!=s2[c-1]);
      if ( iDist < dDist ) cost(r,c) = iDist < mDist ? iDist : mDist;
      else  cost(r,c) = dDist < mDist ? dDist : mDist;
    }
  }
  return cost(r,c);
}

// [[Rcpp::export]]
int edit_distance(string s1, string s2) {
  int r = (int)s1.size();
  int c = (int)s2.size();
  cout<<c;
  IntegerMatrix cost(r+1,c+1);
  fill(cost.begin(), cost.end(), -1.0);
  return edist_dp(s1, s2, cost, r, c);
}
```

```
edit_distance("FOOD","MONEY")
```

```
## [1] 4
```

```
edit_distance("ALGORITHM","ALTRUISTIC")
```

```
## [1] 6
```