Optimal Fish Passage Barrier Removal - Revisited

Appendix A: OPL Model File

```
/*********************************************
* OPL 12.5 Model  * Author: sk
* Creation Date: 11 Oct 2013 at 14:33:56
*********************************************/

/*------------------------------------------
PARAMETER & SET NOTATION
------------------------------------------*/
//available budget
float b = ...;
//number of restoration targets
int ntargets = ...;
//range of restoration targets indexed by t
range T = 1..ntargets;
//target objective weights
float w[T] = ...;
//set of all barrier IDs indexed by j
{string} J = ...;
//downstream ID for barrier j, "NA" if none exists
string dsid[J] = ...;
//upstream habitat at barrier j for target t
float ushab[J][T] = ...;
//current passability of barrier j for target t
float prepass[J][T] = ...;
//number of mitigation projects available at barrier j
int nproj[J] = ...;
//mitigation project data structure
tuple project {
  string barid;      //barrier ID of given mitigation project
  float cost; //cost of mitigation project
  float postpass[T]; //array of post-mitigation passabilities for each target
}
//set of all mitigation projects
{project} A = ...;
//set of all artificial barriers
{string} Jart = {j | j in J: nproj[j] > 0};
```

```
/*-----------------------------------------
DECISION VARIABLES
-----------------------------------------*/
//project mitigation variables: 1 if project i selected, 0 otherwise
dvar boolean x[A];
//change in cumulative passability for target t given implementation of project i
dvar float+ y[A][T];
//cumulative passability at barrier j for target t
dvar float+ z[J][T];

/*-----------------------------------------
OBJECTIVE
-----------------------------------------*/
maximize sum(t in T) w[t] * sum(j in J) ushab[j][t] * z[j][t];

/*-----------------------------------------
CONSTRAINTS
-----------------------------------------*/
subject to{
  //budget constraint
  budget:
    sum(i in A) i.cost * x[i] <= b;

  //maximum of 1 mitigation project selected per barrier
  max_1_proj:
    forall(j in Jart)
      sum(i in A: i.barid == j) x[i] <= 1;

  //flow-balance constraints for barriers without a downstream barrier
  flow_balance_root:
    forall(j in J: dsid[j] == "NA", t in T)
      z[j][t] == prepass[j][t] + sum(i in A: i.barid == j) y[i][t];

  //flow-balance constraints for barriers with a downstream barrier
  flow_balance_branch:
    forall(j in J: dsid[j] != "NA", t in T)
      z[j][t] == prepass[j][t] * z[dsid[j]][t] + sum(i in A: i.barid == j) y[i][t];

  //1st upper bound on increase in cumulative passability
  flow_bounds1:
    forall(j in Jart, i in A: i.barid == j, t in T)
      y[i][t] <= (i.postpass[t] - prepass[j][t]) * x[i];

  //2nd upper bound on increase in cumulative passability
  flow_bounds2:
    forall(j in Jart: dsid[j] != "NA", i in A: i.barid == j, t in T)
      y[iz[j][t] == prepass[j][t] * z[dsid[j]][t] + sum(i in A: i.barid == j) y[i][t];
}
```

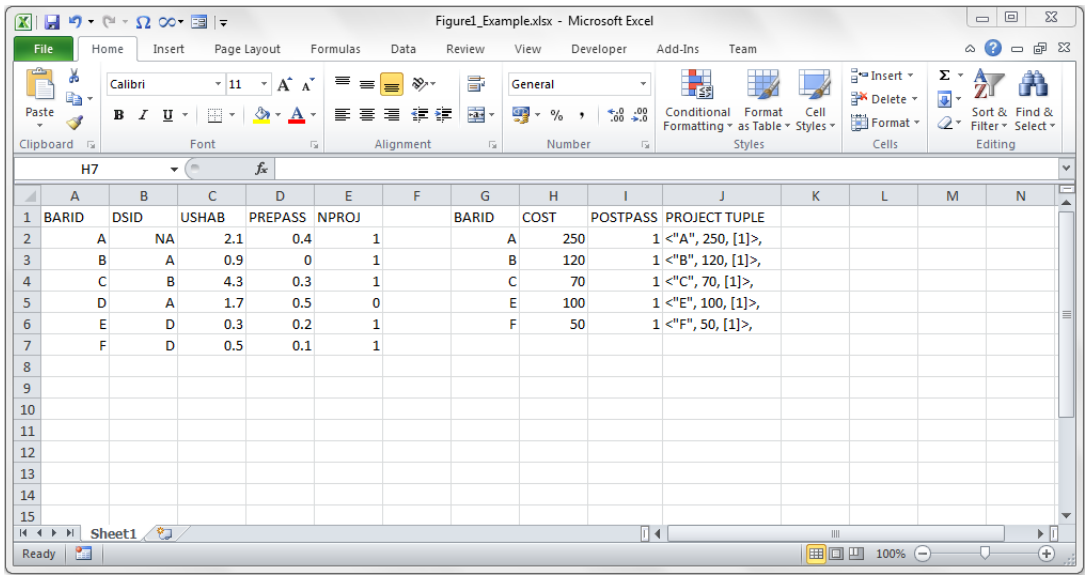
Appendix B: OPL Data File

Example barrier network shown in Figure 1

```
/*********************************************
* OPL 12.5 Data
* Author: sk
* Creation Date: 3 Jun 2013 at 14:06:55
*********************************************/

b = 200.0;
ntargets = 1;
w = [1.0];

SheetConnection data("Figure1_Example.xlsx"); //establish connection with spreadsheet
J from SheetRead(data, "Sheet1!A2:A7"); //read in barrier IDs
dsid from SheetRead(data, "Sheet1!B2:B7"); //read in downstream barrier IDs
ushab from SheetRead(data, "Sheet1!C2:C7");  //read in amount of habitat above each barrier
prepass from SheetRead(data, "Sheet1!D2:D7"); //read in barrier passabilities
nproj from SheetRead(data, "Sheet1!E2:E7"); //read in number of mitigation projects

//mitigation projects
A = {
  <"A", 250, [1]>,
  <"B", 120, [1]>,
  <"C", 70, [1]>,
  <"E", 100, [1]>,
  <"F", 50, [1]>
};
```

Appendix C: Excel Spreadsheet

Example barrier network shown in Figure 1



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	BARID	DSID	USHAB	PREPASS	NPROJ		BARID	COST	POSTPASS	PROJECT TUPLE				
2	A	NA	2.1	0.4	1		A	250	1	<"A", 250, [1]>,				
3	B	A	0.9	0	1		B	120	1	<"B", 120, [1]>,				
4	C	B	4.3	0.3	1		C	70	1	<"C", 70, [1]>,				
5	D	A	1.7	0.5	0		E	100	1	<"E", 100, [1]>,				
6	E	D	0.3	0.2	1		F	50	1	<"F", 50, [1]>,				
7	F	D	0.5	0.1	1									
8														
9														
10														
11														
12														
13														
14														
15														

Figure D1: Screen shot of Figure1_Example.xlsx.

In OPL, tuples that include arrays cannot be read in directly from Excel. As a work around, each of the mitigation project tuples in column J of Figure D1 can be populated using the CONCATENATE function as follows:

=CONCATENATE("<", CHAR(34), G2, CHAR(34), ", ", H2, ", [", I2, "] >,")

The values in the range J2:J6 can subsequently be copied and pasted into an OPL data file. Note that after copying these values into the curly brackets of set "A" in the OPL data file, the final comma in cell J6 should be deleted (see Appendix B above).