



<code>--enable-debug</code>	compile all projects with debug options
<code>--enable-debug-symphony</code>	compile only SYMPHONY project with debug options
<code>--enable-msvc</code>	Under MSys2, compile so that executables can be built on Windows
<code>--enable-static</code>	build static libraries
<code>--enable-static-executable</code>	create a complete static executable (not a DLL)
<code>--enable-gnu-packages</code>	compile with GNU packages
<code>--enable-interactive-optimizer</code>	compile interactive optimizer with debug options
<code>--disable-cgl-cuts</code>	disable generic cut generation
<code>--enable-sensitivity-analysis</code>	compile in the sensitivity analysis module
<code>--enable-root-only</code>	process only the root node
<code>--enable-frac-branching</code>	compile in the fractional branching module
<code>--enable-tests</code>	perform additional sanity checks (if time permits)
<code>--enable-tm-tests</code>	perform more tests
<code>--enable-trace-path</code>	additional debugging options
<code>--enable-cut-check</code>	additional debugging options
<code>--enable-statistics</code>	additional statistics
<code>--enable-pseudo-costs</code>	enable some experimental pseudo-costs
<code>--enable-draw-graph</code>	enable IGD graph drawing application
<code>--with-XXX-incdir</code>	specify the directory with the header files for XXX where XXX is one of LP solver packages: cgl, cplex, xpress
<code>--with-XXX-lib</code>	specify the flags to link with the library for XXX where XXX is one of LP solver packages: cgl, cplex, xpress
<code>--with-lp-solver[=lpsolver]</code>	specify the LP solver in small letters
<code>--with-application</code>	compile the application library
<code>--enable-openmp</code>	compile in OpenMP features
<code>--with-pvm</code>	compile in parallel architecture (as long as PVM is installed and the variable PVM_ROOT is set)
<code>--without-cg</code>	compile without cut generator module
<code>--without-cp</code>	compile without cut pool module
<code>--without-lp</code>	compile without LP solver module
<code>--without-tm</code>	compile without tree manager module

```
[frame=lines]
```

```
int main(int argc, char **argv)
```

```
{
```

```
    sym_environment *env = sym_open_environment();
```

```
    sym_parse_command_line(env, argc, argv);
```

```
    sym_load_problem(env);
```

```
    sym_solve(env);
```

```
    sym_close_environment(env);
```

```
[frame=lines]
```

```
int main(int argc, char **argv)
```

```
{
```

```
    sym_environment *env = sym_open_environment();
```

```
    sym_parse_command_line(env, argc, argv);
```

```
    sym_load_problem(env);
```

```
    sym_set_int_param(env, "find_first_feasible", TRUE);
```

```
    sym_set_int_param(env, "node_selection_strategy", DEPTH_FIRST);
```

```
    sym_solve(env);
```

```
    sym_set_int_param(env, "find_first_feasible", FALSE);
```

```
    sym_set_int_param(env, "node_selection_strategy", BEST_FIRST);
```

```
    sym_warm_solve(env);
```

```
[frame=lines]
```

```
int main(int argc, char **argv)
{
    warm_start_desc *ws;
    sym_environment *env = sym_open_environment();
    sym_parse_command_line(env, argc, argv);
    sym_load_problem(env);
    sym_set_int_param(env, "node_limit", 100);
    sym_set_int_param(env, "keep_warm_start", TRUE);
    sym_solve(env);
    ws = sym_get_warm_start(env);
    sym_set_int_param(env, "node_limit", -1);
    sym_warm_solve(env);
    sym_set_obj_coeff(env, 0, 100);
    sym_set_obj_coeff(env, 200, 150);
    sym_set_warm_start(ws);
    sym_warm_solve(env);
}
```





```
[frame=lines]
```

```
int main(int argc, char **argv)
```

```
{
```

```
    sym_environment *env = sym_open_environment();
```

```
    sym_parse_command_line(env, argc, argv);
```

```
    sym_load_problem(env);
```

```
    sym_set_obj2_coeff(env, 0, 1);
```

```
    sym_mc_solve(env);
```



```
[frame=lines]
int main(int argc, char **argv)
{
    OsiSymSolverInterface si;
    si.parseCommandLine(argc, argv);
    si.loadProblem();
    si.branchAndBound();
}
```



ERIS







2020-2021

WORLDWIDE

Bounding Operation

Input: A subproblem \mathcal{S} , described in terms of a “small” set of inequalities $\mathcal{S} = \{x^s : s \in \mathcal{F} \text{ and } ax^s \leq \beta \ \forall (a, \beta) \in \mathcal{L}'\}$ and α , an upper bound on the optimal value.

Output: Either (1) an optimal solution $s^* \in \mathcal{S}$ to the subproblem, (2) the optimal value of the subproblem, or (3) a message **pruned** indicating the subproblem should not be considered further.

Step 1. Set $\mathcal{C} \leftarrow \mathcal{L}'$.

Step 2. Solve the LP $\min\{cx : ax \leq \beta \ \forall (a, \beta) \in \mathcal{C}\}$.

Step 3. If the LP has a feasible solution \hat{x} , then go to Step 4. Otherwise, output **pruned**. This subproblem has no feasible solutions.

Step 4. If $c\hat{x} < \alpha$, then go to Step 5. Otherwise, STOP and output the optimal value of the subproblem cannot produce a solution of value better than α .

Step 5. If \hat{x} is the incidence vector of some $\hat{s} \in \mathcal{S}$, then \hat{s} is the optimal solution to this subproblem. STOP and output \hat{s} as s^* . Otherwise, apply separation heuristics to \hat{x} to get a set of violated inequalities \mathcal{C}' . If $\mathcal{C}' = \emptyset$, then return the value of an optimal element of \mathcal{S} . STOP and return \hat{x} and its value. Otherwise, set $\mathcal{C} \leftarrow \mathcal{C} \cup \mathcal{C}'$ and go to Step 2.















Handwritten text in a cursive script, likely a signature or a name, rendered in black ink on a white background. The text is highly stylized and difficult to decipher, but appears to be a single line of writing.

Branching Operation

Input: A subproblem \mathcal{S} and \hat{x} , the LP solution yielding the lower bound.

Output: S_1, \dots, S_p such that $\mathcal{S} = \cup_{i=1}^p S_i$.

Step 1. Determine sets $\mathcal{L}_1, \dots, \mathcal{L}_p$ of inequalities such that $\mathcal{S} = \{x \in \mathbb{R}^n : ax \leq \beta \ \forall (a, \beta) \in \mathcal{L}_i\}$ and $\hat{x} \notin \cup_{i=1}^n S_i$.

Step 2. Set $S_i = \{x \in \mathcal{S} : ax \leq \beta \ \forall (a, \beta) \in \mathcal{L}_i \cup \mathcal{L}'\}$ where \mathcal{L}' is the set of inequalities used to describe \mathcal{S} .

Generic Branch and Cut Algorithm

Input: A data array specifying the problem instance.

Output: The global optimal solution s^* to the problem instance.

Step 1. Generate a “good” feasible solution \hat{s} using heuristics. Set $\hat{c} = c\hat{s}$.

Step 2. Generate the first subproblem \mathcal{S}^I by constructing a small subproblem \mathcal{S} that is valid for \mathcal{P} . Set $A \leftarrow \{\mathcal{S}^I\}$.

Step 3. If $A = \emptyset$, STOP and output \hat{s} as the global optimum s^* . Otherwise, select $\mathcal{S} \in A$. Set $A \leftarrow A \setminus \{\mathcal{S}\}$. Process \mathcal{S} .

Step 4. If the result of Step 3 is a feasible solution \bar{s} , then $c\bar{s} < c\hat{s}$. Set $\hat{s} = \bar{s}$ and $\hat{c} = c\bar{s}$, and go to Step 3. If the subproblem was pruned, go to Step 3. Otherwise, go to Step 5.

Step 5. Perform the branching operation. Add the set of subproblems A to A and go to Step 3.

$$\begin{array}{ll}
 \text{vmin} & [cx, dx], \\
 \text{s.t.} & Ax \leq b, \\
 & x \in \mathbb{Z}^n.
 \end{array}$$











100 + 100 = 200











max $\{ \alpha(\beta - 1) - \alpha(\beta - 1) \}$.









C++ Interface	C Interface	Description
OsiSymSolverInterface	sym_open_environment	create a new environment.
loadProblem	sym_load_problem	load the problem read trough an MIP file.
branchAndBound	sym_solve/sym_warm_solve	solve the MILP problem from scratch or from a warm start if loaded.
resolve	sym_warm_solve	re-solve the MILP problem after some changes.
initialSolve	sym_solve	solve the MILP problem from scratch.
multiCriteriaBranchAndBound	sym_mc_solve	solve the multi criteria problem.
setInitialData	sym_set_defaults	set the parameters to their defaults.
parseCommandLine	sym_parse_command_line	read the command line arguments.
findInitialBounds	sym_find_initial_bounds	find the initial bounds via the user coefficients.
createPermanentCutPools	sym_create_permanent_cut_pools	save the global cuts.
loadProblem	sym_explicit_load_problem	load the problem through a set of arrays.
getWarmStart	sym_get_warm_start	get the warm start description.
setWarmStart	sym_set_warm_start	set the warm start description.
getLbForNewRhs	sym_get_lb_for_new_rhs	find a lower bound to the new rhs values using the post solution info.
getUbForNewRhs	sym_get_lb_for_new_rhs	find an upper bound to the new rhs values using the post solution info.
getLbForNewObj	sym_get_lb_for_new_rhs	find a lower bound to the new obj value using the post solution info.
getUbForNewObj	sym_get_lb_for_new_rhs	find an upper bound to the new obj value using the post solution info.
reset	sym_close_environment	return the allocated memory.
setIntParam	sym_set_int_param	set the integer type OSI parameter.
setSymParam(int)	sym_set_int_param	set the integer type SYMPHONY parameter.
setDblParam	sym_set_dbl_param	set the double type OSI parameter.
setSymParam(double)	sym_set_dbl_param	set the double type SYMPHONY parameter.
setStrParam	sym_set_str_param	set the string type OSI parameter.
setSymParam(string)	sym_set_str_param	set the string type SYMPHONY parameter.
getIntParam	sym_get_int_param	get the value of the integer type OSI parameter.
getSymParam(int &)	sym_get_int_param	get the value of the integer type SYMPHONY parameter.
getDblParam	sym_get_dbl_param	get the value of the double type OSI parameter.
getSymParam(double &)	sym_get_dbl_param	get the value of the double type SYMPHONY parameter.
getStrParam	sym_get_str_param	get the value of the string type OSI parameter.
getSymParam(string &)	sym_get_str_param	get the value of the string type SYMPHONY parameter.
isProvenOptimal	sym_is_proven_optimal	query the problem status.
isProvenPrimalInfeasible	sym_is_proven_primal_infeasible	query the problem status.
isPrimalObjectiveLimitReached	sym_is_target_gap_achieved	query the problem status.
isIterationLimitReached	sym_is_iteration_limit_reached	query the problem status.
isTimeLimitReached	sym_is_time_limit_reached	query the problem status.
isTargetGapReached	sym_is_target_gap_achieved	query the problem status.
getNumCols	sym_get_num_cols	get the number of columns.
getNumRows	sym_get_num_rows	get the number of rows.
getNumElements	sym_get_num_elements	get the number of nonzero elements.
getColLower	sym_get_col_lower	get the column lower bounds.
getColUpper	sym_get_col_upper	get the column upper bounds.
getRowSense	sym_get_row_sense	get the row senses.
getRightHandSide	sym_get_rhs	get the rhs values.
getRowRange	sym_get_row_range	get the row range values.
getRowLower	sym_get_row_lower	get the row lower bounds.
getRowUpper	sym_get_row_upper	get the row upper bounds.
getObjCoefficients	sym_get_obj_coeff	get the objective function vector.

C++ Interface	C Interface	Description
getObjSense	sym_get_obj_sense	get the objective sense.
isContinuous	sym_is_continuous	query the variable type.
isBinary	sym_is_binary	query the variable type.
isInteger	sym_is_integer	query the variable type.
isIntegerNonBinary	-	query the variable type.
isFreeBinary	sym_is_binary	query the variable type.
getMatrixByRow	-	get the constraint matrix by row orientation.
getMatrixByCol	-	get the constraint matrix by column orientation.
getInfinity	-	get the infinity definition of SYMPHONY.
getColSolution	sym_get_col_solution	get the current best column solution.
getRowActivity	sym_get_row_activity	get the current row activity.
getObjValue	sym_get_obj_val	get the current best objective value.
getPrimalBound	sym_get_primal_bound	get the primal upper bound.
getIterationCount	sym_get_iteration_count	get the number of the analyzed tree nodes.
setObjCoeff	sym_set_obj_coeff	set the objective function vector.
setObj2Coeff	sym_set_obj2_coeff	set the second objective function vector.
setColLower	sym_set_col_lower	set the column lower bounds.
setColUpper	sym_set_col_upper	set the column upper bounds.
setRowLower	sym_set_row_lower	set the row lower bounds.
setRowUpper	sym_set_row_upper	set the row upper bounds.
setRowType	sym_set_row_type	set the row characteristics.
setObjSense	sym_set_obj_sense	set the objective sense.
setColSolution	sym_set_col_solution	set the current solution.
setContinuous	sym_set_continuous	set the variable type.
setInteger	sym_set_integer	set the variable type.
setColName	sym_set_col_names	set the column names.
addCol	sym_add_col	add columns to the constraint matrix.
addRow	sym_add_row	add rows to the constraint matrix.
deleteCols	sym_delete_cols	delete some columns from the constraint matrix.
deleteRows	sym_delete_rows	delete some rows from the constraint matrix.
writeMps	-	write the current problem in MPS format.
applyRowCut	-	add some row cuts.
applyColCut	-	add some column cuts.
SymWarmStart(warm_start_desc *)	sym_create_copy_warm_start	create a SYMPHONY warm start by copying.
SymWarmStart(char *)	sym_read_warm_start	create a SYMPHONY warm start reading from a file.
getCopyOfWarmStartDesc	sym_create_copy_warm_start	get the copy of the warm start structure.
writeToFile	sym_write_warm_start_desc	write the loaded warm start to a file.













12 + 12 = 24



















$$s_z = a \times \min\{z_+, z_-\} + (1 - a) \times \max\{z_+, z_-\}$$



0.1





C++ Interface	C Interface	Value
OsiSymVerbosity	verbosity	-user
OsiSymWarmStart	warm_start	-boolean
OsiSymNodeLimit OsiMaxNumIteration OsiMaxNumIterationHotStart	node_limit	-user
OsiSymFindFirstFeasible	find_first_feasible	-boolean
OsiSymSearchStrategy	node_selection_rule	LOW HIGH BREAK DEPTH
OsiSymUsePermanentCutPools	use_permanent_cut_pools	-boolean
OsiSymGenerateCglGomoryCuts	generate_cgl_gomory_cuts	-boolean
OsiSymGenerateCglKnapsackCuts	generate_cgl_knapsack_cuts	-boolean
OsiSymGenerateCglOddHoleCuts	generate_cgl_oddhole_cuts	-boolean
OsiSymGenerateCglProbingCuts	generate_cgl_probing_cuts	-boolean
OsiSymGenerateCglCliqueCuts	generate_cgl_clique_cuts	-boolean
OsiSymGenerateCglFlowAndCoverCuts	generate_cgl_flow_and_cover_cuts	-boolean
OsiSymGenerateCglRoundingCuts	generate_cgl_rounding_cuts	-boolean
OsiSymGenerateCglLiftAndProjectCuts	generate_cgl_lift_and_project_cuts	-boolean
OsiSymKeepWarmStart	keep_warm_start	-boolean
OsiSymTrimWarmTree	trim_warm_tree * -boolean-	
OsiSymDoReducedCostFixing	do_reduced_cost_fixing	-boolean
OsiSymMCFindSupportedSolutions	mc_find_supported_solutions	-boolean
OsiSymSensitivityAnalysis	sensitivity_analysis	-boolean
OsiSymRandomSeed	random_seed	-user
OsiSymDivingStrategy	diving_strategy	BEST COM COM
OsiSymDivingK	diving_k	-user
OsiSymDivingThreshold	diving_threshold	-user
OsiSymGranularity	granularity	-user
OsiSymTimeLimit	time_limit	-user
OsiSymGapLimit	gap_limit	-user
OsiObjOffset	-	-user
OsiProbName	problem_name	-user