Mixed Integer Programming (MIP) in EnCompass

EnCompass utilizes mixed integer programming to determine the optimal solution to capacity expansion, unit commitment, and economic dispatch problems.

Economic Dispatch

The economic dispatch problem seeks to minimize total production costs given a commitment schedule of which units are online and offline in every interval (usually one hour). EnCompass formulates this as a linear problem by using a piecewise-linear representation of unit heat and emission rates, and either a zonal or DC (linearized) powerflow model for transmission. Constraints applied for the economic dispatch include load and ancillary service requirements, transmission limits, fuel limits, environmental limits, storage limits and efficiencies, capacity factor (energy) limits, ramp rates, and resource capacity limits for energy and ancillary services.

Linear programming is a fast, robust, and well-established process that will always return an optimal solution if the problem is feasible (i.e., the constraints are not conflicting). EnCompass uses "soft" constraints for load balance, ancillary services, and certain transmission limits by allowing the limits to be violated subject to input penalties (unserved load and curtailment penalties). In this way, the problem will always be feasible, and any limit violations are reported. In most cases, there is only a single optimal solution. However, if there are multiple units with identical costs, the selection of which units to dispatch is arbitrary. EnCompass will always produce the exact same solution for the same scenario. If a unit that was not dispatched is removed from the scenario, the structure of the problem changes, and a different dispatch of identical units could occur if a different route were taken to find an optimal solution.

When EnCompass is run using the "No Commitment" option, the minimum capacities of resources that are not must-run are relaxed, so that there is no unit commitment problem to solve. In this mode, any startup and no-load costs are converted to linear $/MWh costs using the input Expected Runtime (or if not set, the Minimum Uptime), and are added to a unit's energy and ancillary service costs. This option is the fastest way in which to run EnCompass.

Unit Commitment

The unit commitment problem extends the economic dispatch problem by allowing the selection of which units are online and offline in every interval, given a set of units with fixed commission and retirement dates. This selection uses integer, or whole-number, variables together with the continuous variables from the economic dispatch problem, which is why the methodology is referred to as a "mixed" integer program.

The unit commitment constraints that EnCompass applies includes minimum uptime, minimum downtime, maximum daily and weekly starts, and profiles for which intervals are allowed for unit starts and shutdowns. Fuel requirements and direct costs can be applied to starts and shutdowns, with the option to vary startup requirements based on cold, warm, and hot input definitions. Operating constraints can be applied across a group of units to model load pocket and voltage support requirements, as well as dependencies and other restrictions.

When EnCompass is run using the "Partial Commitment" option, all of the unit commitment costs and constraints are applied, but the number of units committed in any interval is allowed to be a continuous variable between 0 and 1. For example, if the optimal solution included a value of 0.3 for the number of units committed, this would incur 30% of the cost of a start and only allow the unit to dispatch up to 30% of its capacity. The unit would still have to be at least 30% committed for the minimum uptime, and once it goes below that cannot increase until the minimum downtime has passed. The Partial Commitment option turns the unit commitment problem into a linear problem, which makes it faster to solve than the "Full Commitment" option and provides more detail and constraints than the "No Commitment" option.

Capacity Expansion

The capacity expansion problem extends the unit commitment and economic dispatch problem by allowing the selection of new resources, transmission upgrades, and economic retirements. This selection uses also uses integer variables that represent the number of resource additions and retirements in each year. The economic carrying charge is used to represent capital costs for new projects, which increases at the rate of construction escalation and provides the same present value of annual revenue requirements over the book life.

Instead of firm reserve margin constraints, EnCompass uses capacity demand curves to incent meeting reserve margin targets. These can represent "high cliffs" where the penalty for falling short of the target reserve margin is very high ($10,000/kW-year) and then goes to 0 once the reserve margin is met; or they can be downward-sloping curves like those used in PJM, New York and New England for capacity markets.

Each project can have constraints on the maximum additions (incremental) per year, and the minimum and maximum active (cumulative) projects each year. Project Constraints can be used to set these constraints over a group of projects, which can include exclusivity and dependencies.

EnCompass includes a "Partial" optimization option which will allow the number of additions to be a continuous variable. For example, if the optimal solution included a value of 0.3 for the number of additions, this would incur 30% of the capital costs and only consider 30% of the capacity added. Over the operating life of the project, the number of active projects would be at least 30%. If the unit commitment option is "No Commitment" or "Partial Commitment" (which are the typical settings for capacity expansion), Partial project option turns the capacity expansion problem into a linear problem, which makes it faster to solve than the "Full" option. There is also a "Rounded" option which will automatically round up all additions and retirement to the next whole number, but this is typically only used for market-based capacity expansion over large regions. Finally, even with the "Full" option, individual projects can consider partial additions after an input year, which improves the overall runtime.

The MIP Process

If either the unit commitment or capacity expansion options are set to "Full", EnCompass will solve the problem using mixed integer programming. Unlike linear programming, it is not always feasible to find the global optimal solution to a mixed integer problem since the process requires evaluating numerous potential integer solutions. Instead, the problem is considered to be solved when the costs of the best integer solution found is within an input tolerance of the cost of the best remaining partial solution

(known as the best bound). The tolerance is measured as the percent difference between the best solution and best bound, and in EnCompass the MIP Stop Basis is input as basis points (1/100th of 1%).

The MIP process first determines the best partial solution using linear programming, as if the option had been set to "Partial". The cost of this solution then becomes the initial best bound, since rounding partial variables up or down will only increase the costs from there. Then, the MIP will create and evaluate several subproblems to find integer solutions and eliminate other possibilities. When a better solution is found, this reduces the best solution cost; when a path is eliminated, this increases the best bound cost. The process continues until the gap between these two costs is within the input tolerance.

Consider a simple example of a one-year capacity expansion problem with three potential projects (P1, P2, and P3) where each project has a maximum of 1. The first step is to solve the partial problem, and assume it provides these results:

- Node 0: Cost = $15.5 million, P1 = 0.3, P2 = 0.8, P3 = 0

The best bound is now $15.5 million, and the MIP will now start to evaluate the subproblems by branching on the partial solutions. For example, two subproblems will be created, one with the constraint P1 = 0 and the other with the constraint P1 = 1. These subproblems are then solved using linear programming, and assume these results:

- Node 1: Cost = $16.1 million, P1 = 0, P2 = 1, P3 = 0
- Node 2: Cost = $15.8 million, P1 = 1, P2 = 0.4, P3 = 0.1

Note that the project results for Node 1 are now all integers, and we have our first feasible solution. Node 2 still has partial projects, so the best bound now increases to $15.8 million. The gap between the best solution and best bound is 1.9%. If the MIP Stop Basis was set to 200, the process will stop and return Node 1 as the best solution. Assume that the MIP Stop Basis is lower, and the process will now branch on Node 2 by setting P2 = 0 and P2 = 1:

- Node 3: Cost = $15.9 million, P1 = 1, P2 = 0, P3 = 0.3
- Node 4: Cost = $16.2 million, P1 = 1, P2 = 1, P3 = 0

Node 4 is a feasible integer solution, but has a higher cost than Node 1, so the process does not do anything else with Node 4 (that "branch has been pruned"). Node 3 is a partial solution, so the best bound increases to $15.9 million, leaving a gap of 1.3% with the best solution (Node 1). Assume that the MIP Stop Basis is less than 120, so the process will now branch on Node 3 by setting P3 = 0 and P3 = 1:

- Node 5: Cost = $15.9 million, P1 = 1, P2 = 0, P3 = 0
- Node 6: Cost = $16.3 million, P1 = 1, P2 = 0, P3 = 1

The process is now left with only integer solutions, so the best bound and best solution are both $15.9 million, the gap is 0%, and Node 5 is the optimal solution.

Objective Functions and the Unified Solution

Models like Strategist and EGEAS use dynamic programming to enumerate all feasible nodes. Each node is run through a non-linear probabilistic sub-module to determine production costs, which are added to the capital costs to determine the selected objective function value. The objective function is then ranked across all those nodes to determine the optimal plan. In this simple problem, there were 2 x 2 x 2

= 8 nodes to evaluate with dynamic programming, and 7 nodes with mixed integer programming. In a typical multi-year expansion problem, there are usually thousands of integer variables that can take values larger than 1. This makes the number of nodes to evaluate with a dynamic program skyrocket and requires additional constraints to be imposed to bring that number down. With mixed integer programming, the number of nodes is manageable since only those nodes that show promise are evaluated and used to look for other nodes.

With dynamic programming, the objective function can be distorted between the production cost sub-module and the capacity expansion decision. For example, if the objective function is to minimize total utility costs plus emission externalities, the ranking of nodes may pick up externality costs from the production cost sub-module, even if that was not included in the commitment and dispatch objective function. With mixed integer programming, there is no decoupling of capacity expansion, unit commitment, and economic dispatch, so all three of these decisions work together to minimize the single selected objective function. As an example, given the objective function of minimizing total utility costs plus emission externalities, a low-cost alternative might be to displace a MWh from a higher emitting resource, such as a coal-fired unit, with a MWh from a lower emitting unit such as either renewable or gas-fired generation. Depending on the design of the model, a dynamic programming algorithm might recognize one, both or neither of these options and possibly not produce the most economical alternative. Conversely, a mixed integer model that co-optimizes production cost and capacity expansion will evaluate all options for minimizing the objective function.

To illustrate this further, assume a fourth potential project, P4, is considered. The dynamic programming approach builds a decision tree, with a branch for every project decision (add 0 or 1). The result is 16 feasible solutions, shown in the figure below as orange leaves on the tree, each of which must be evaluated with the production cost sub-module:
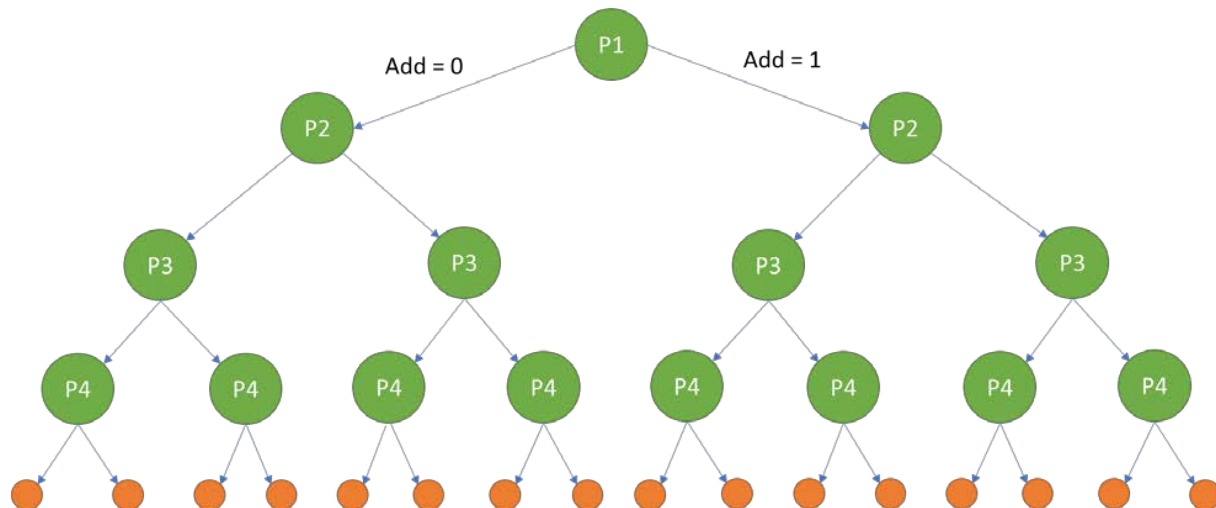
Because the mixed integer process uses a unified solution, it knows the change in costs as the tree is being built and can prune branches that will always produce higher-cost feasible solutions. In the figure below, those pruned branches are shown in black, and the nodes from the example above are identified:
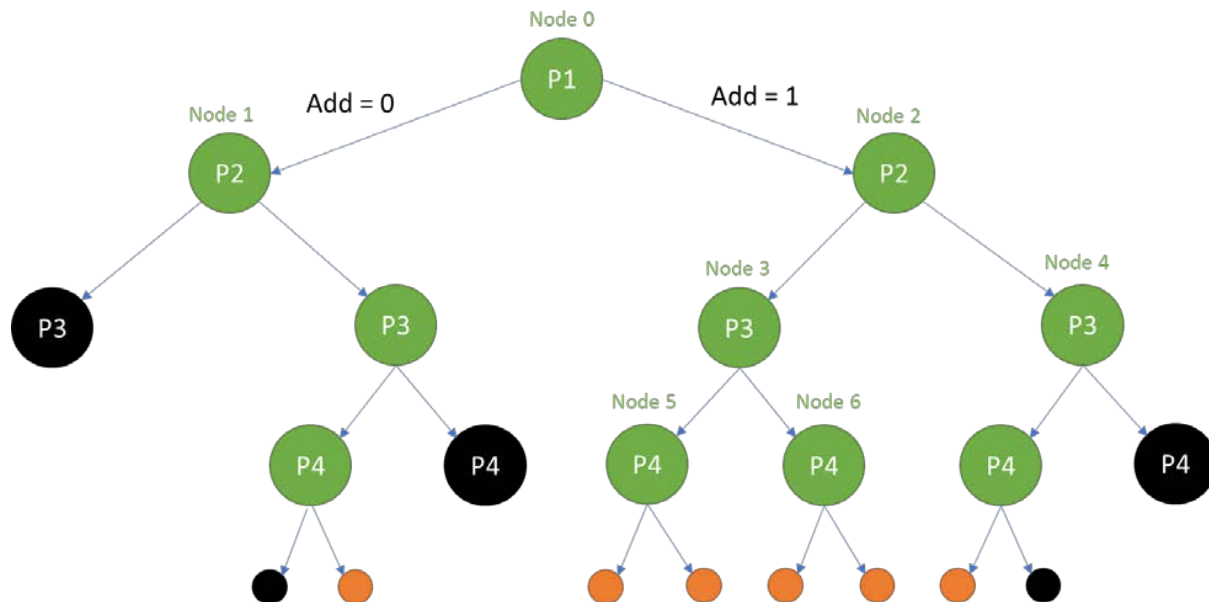
Another key advantage is that in the MIP process, each node can be evaluated using the solution to the prior node as a starting point, greatly reducing the processing time required to evaluate new nodes. The non-linear production cost sub-module of the dynamic program cannot "learn" as it goes, and each feasible solution must be evaluated from scratch.

To minimize the size of the problem that must be solved, EnCompass does not include the variables, constraints, and costs of any decisions which are fixed. This means that if the selections of one project in a capacity expansion optimization is "frozen" and the case is run again, the objective function values will be lower since the capital and fixed operating costs of that frozen project are not included. Since the convergence threshold is a percentage of the objective function, that gap becomes tighter, and a different overall plan with a slightly lower cost may be chosen.

The structure of the problem can also impact the selection of which variables are branched and the path that is used to find solutions. For example, removing limits that are never binding or resources and projects that are never utilized does not change the underlying economics, but it does make the problem smaller, which could lead to different approaches and different solutions that are both within the convergence tolerance. For capacity expansion problems where the MIP Stop Basis is set to a low value like 50 (0.5%), multiple solutions that are within that threshold should be considered to have comparable costs over the multi-year optimization period.

Xpress Optimization Suite

EnCompass uses the Xpress Optimization Suite from FICO to solve the linear and mixed integer problems described above. The branch-and-bound process can be sped up considerably by making better choices on which variables to branch on, and by performing heuristic searches for additional nodes. Xpress uses

these techniques and others to provide the best possible performance for solving mixed integer problems.

One of the key techniques to reduce runtime for large problems is parallelism. This allows multiple "leaf" nodes to be solved simultaneously, based on the number of available computing cores and memory. As a result, the solution path may be different when solving using one set of computing resources versus another. This could produce two different solutions that are both feasible and within the input gap threshold.