Feature	Description	Count	Reference
Static Features (18)			
Objective function coeffs	Value of the coefficient (raw, positive only, negative only)	3	
Num. constraints	Number of constraints that the variable participates in (with a non-zero coefficient)	1	
Stats. for constraint degrees	The *degree of a constraint* is the number of variables that participate in it. A variable may participate in multiple constraints, and statistics over those constraints' degrees are used. The constraint degree is computed on the root LP (mean, stdev., min, max)	4	
Stats. for constraint coeffs.	A variable's positive (negative) coefficients in the constraints it participates in (count, mean, stdev., min, max)	10	
Dynamic Features (54)			
Slack and ceil distances	$\min\{\tilde{x}_j - \lfloor \tilde{x}_j \rfloor, \lceil \tilde{x}_j \rceil - \tilde{x}_j\}$ and $\lceil \tilde{x}_j \rceil - \tilde{x}_j$	2	
Pseudocosts	Upwards and downwards values, and their corresponding ratio, sum and product, weighted by the fractionality of x_j	5	(Achterberg 2009)
Infeasibility statistics	Number and fraction of nodes for which applying SB to variable x_j led to one (two) infeasible children (during data collection)	4	
Stats. for constraint degrees	A dynamic variant of the static version above. Here, the constraint degrees are on the current node's LP. The ratios of the static mean, maximum and minimum to their dynamic counterparts are also features	7	
Min/max for ratios of constraint coeffs. to RHS	Minimum and maximum ratios across positive and negative right-hand-sides (RHS)	4	(Alvarez, Louveaux, and Wehenkel 2014)
Min/max for one-to-all coefficient ratios	The statistics are over the ratios of a variable's coefficient, to the sum over all other variables' coefficients, for a given constraint. Four versions of these ratios are considered: positive (negative) coefficient to sum of positive (negative) coefficients	8	(Alvarez, Louveaux, and Wehenkel 2014)
Stats. for active constraint coefficients	An active constraint at a node LP is one which is binding with equality at the optimum. We consider 4 weighting schemes for an active constraint: unit weight, inverse of the sum of the coefficients of all variables in constraint, inverse of the sum of the coefficients of only candidate variables in constraint, dual cost of the constraint. Given the absolute value of the coefficients of x_j in the active constraints, we compute the sum, mean, stdev., max. and min. of those values, for each of the weighting schemes. We also compute the weighted number of active constraints that x_j is in, with the same 4 weightings	24	(Patel and Chinneck 2007)

Table 1: Description of the atomic features.

Experimental Results

Setup. We use the C API of IBM ILOG CPLEX 12.6.1 to implement various strategies using control callbacks, in single-thread mode. To evaluate the performance of any variable selection strategy $\mathcal{A}$, the strategy is run on a set of instances with a time cut-off of t_{max} seconds. An instance $\mathcal{I}$ is *solved* by strategy $\mathcal{A}$ if and only if the run terminates within the tolerance gaps (we use default CPLEX values). If an instance $\mathcal{I}$ is not solved by the time cut-off, it is referred to as *unsolved*. All experiments were run on a cluster of four 64-core machines with AMD 2.4 GHz processors and 264 GB of memory; each run was limited to 2 GB of memory, and no run failed for memory reasons.

To isolate the effects of changing the variable selection strategy, we provide the optimal value as upper cutoff to CPLEX before the start of the search. This measure reduces the effect of node selection on the search, as the primal bound is given by the upper cutoff, and the order in which nodes are expanded has little impact on the tree itself. Additionally, cuts are allowed at the root only, and primal heuristics are disabled. These measures are common in branching studies (Linderoth and Savelsbergh 1999; Fischetti and Monaci 2012; Karzan, Nemhauser, and Savelsbergh 2009), since they eliminate the interference between variable selection and other components of the solver, such as node selection. This also reduces *performance variability*, which we discuss in the next section.

Instances. We use the "Benchmark" set from MIPLIB2010 as our test set; we refer to (Koch et al. 2011) for details. This set was designed to span a variety of problem classes, applications, dimensions, levels of difficulty, etc., and is routinely used for evaluating branching strategies. The "Benchmark" set consists of 87 instances that can be solved by at least one commercial solver within 2 hours on a high-end PC. Note that since we turn off multi-threading and cuts beyond the root, we cannot expect to solve all instances within 2 hours. Hence, we set the time cut-off t_{max} to 5 hours (18,000 seconds). Three infeasible instances are excluded.

For each of the 84 instances we consider, we run every strategy with 10 different random seeds, for every variable selection strategy. Recent studies have shown that MIP solvers can be very sensitive to seemingly performance-neutral perturbations to their inputs (Lodi and Tramontani 2013; Achterberg and Wunderling 2013). Therefore, runs with different seeds are necessary for obtaining meaningful results. In CPLEX, such perturbations can be induced by changing CPLEX's internal random seed via its C API.

Branching strategies. We experiment with five strategies. CPLEX-D is the strategy that branches on the variable chosen by the solver with its default variable selection rule (as set by `CPX_PARAM_VARSEL`); this is done within a callback, as for all other strategies. Up until 2013, CPLEX developers report that the default selection rule is "a version of *hybrid branching*" (Achterberg and Wunderling 2013). SB refers to Strong Branching, while PC refers to pseudocost branching with SB initialization of the PC values (Linderoth and Savelsbergh 1999). SB+PC is a hybrid of SB for the first