CIP and MIQP Models for the Load Balancing Nurse-to-Patient Assignment Problem

Wen-Yang Ku, Thiago Pinheiro, and J. Christopher Beck

Department of Mechanical & Industrial Engineering
University of Toronto, Toronto, Ontario M5S 3G8, Canada
{wku,jcb}@mie.utoronto.ca, thiagopj@gmail.com

Abstract. The load balancing nurse-to-patient assignment problem requires the assignment of nurses to patients to minimize the variance of the nurses' workload. This challenging benchmark is currently best solved exactly with constraint programming (CP) using the SPREAD constraint and a problem-specific heuristic. We show that while the problem is naturally modelled as a mixed integer quadratic programming (MIQP) problem, the MIQP does not match the performance of CP. We then develop several constraint integer programming (CIP) models that include bounds propagation, linear relaxations, and cutting planes associated with the QUADRATIC, GCC, and SPREAD global constraints. While the QUADRATIC and GCC techniques are known, our additions to the SPREAD constraint are novel. Our empirical results demonstrate that the CIP approach substantially out-performs the MIQP model, but still lags behind CP. Finally, we propose a simple problem-specific variable ordering heuristic which greatly improves the CIP models, achieving performance about an order of magnitude faster than CP and establishing a new state of the art.

1 Introduction

The load balancing nurse-to-patient assignment problem assigns nurses to a hospital zone (e.g., a nursery room) and to patients within the zone to minimize the variance in the total acuity of patients across nurses. A constraint programming approach, combining the SPREAD constraint [1] with problem-specific search heuristics, is currently the state of the art for solving this problem exactly [2,3]. In this paper, we address the problem with two approaches that, to our knowledge, have not yet been applied: mixed integer quadratic programming (MIQP) and constraint integer programming (CIP) [4,5].

Mixed integer linear programming (MILP) techniques have been extended to reason about convex quadratic constraints [6], allowing MIQPs to be solved by commercial MILP solvers. Since variance minimization can be modelled as an MIQP and underlying MILP technology has seen substantial improvement over the past few years [7], we develop and apply an MIQP model to the problem. The model is natural, given the direct representation of the quadratic constraints, however we show that model performs substantially worse using CPLEX than the existing CP model implemented in COMET.

We then improve the MIQP model in two orthogonal directions. First, in the constraint integer programming (CIP) framework, we add global constraints that embed both inference techniques in the form of bounds propagation and relaxation-based reasoning via constraint-specific linear relaxations and cutting planes. These techniques are functional extensions to global constraints (beyond filtering) that can be implemented in any solver that allows linear relaxations and the dynamic addition of cutting planes. In particular, we augment the MIQP to include QUADRATIC [8], GCC, and SPREAD global constraint. For the first two constraints, the propagation, relaxation, and cutting plane techniques are taken from the literature. For the SPREAD constraint, we implement the bound consistency algorithm due to Schaus et al. [9] and develop novel relaxation and cutting plane techniques. Our augmented global constraints can be soundly used in any models in which the un-augmented global constraints appear. The second direction of improvements is to modify the default branching heuristics. Given the structure of the nurse-to-patient assignment problem, we develop two simple, static variable ordering heuristics that influence the default, general branching rules of the CIP solver, SCIP [4,10].

Our empirical results show that the CIP models without problem-specific branching heuristics significantly outperform the MIQP model in CPLEX, achieving about a three times speed-up. However, the CP model is still more than twice as fast. The primary benefit arises from the use of the QUADRATIC constraint with worse overall run-time performance from the additional inclusion of the GCC and SPREAD constraints.

When problem-specific branching priorities are used, all three CIP models outperform the state-of-the-art CP model. The best model includes all three global constraints and results in an order of magnitude improvement over CP. Compared to the default heuristics, the problem-specific heuristics find and prove optimal solutions with 30 to 40 times less effort in terms of both search tree size and run-time. Subsequent analysis shows that the improved performance arises due to the rapid improvement in both the primal (upper) bound and the dual (lower) bound early in the search.

The paper is organized as follows. In Section 2 we present the problem and previous results. We present the CP, MIQP, and CIP models in Section 3. Section 4 describes the global constraints used in the CIP models while Section 5 defines the branching heuristics. Sections 6 and 7 provide computational results and discussions. We conclude in Section 8.

2 Background

The load balancing nurse-to-patient assignment problem requires the assignment of nurses to new-born infant patients across different zones in a hospital [11]. Each infant is hospitalized in one of the rooms, or zones, in the nursery and requires a certain amount of care depending on the acuity of his/her condition. The problem has two main decisions. First, each nurse has to be assigned to a zone in which he/she will work for the entire shift. Second, each nurse has to be assigned to a set of patients in the same zone. The objective is to minimize the variance of the total acuity assigned to each nurse. Such an assignment will avoid overloading the nurses, which can result in stress and poor quality of the care.

The problem was originally modeled and solved as an MILP with a linear objective function that minimizes the sum of the differences between the maximum and minimum assigned workload in each zone [11]. However, although the objective function ensures the workload of each zone is evenly distributed, the workload difference between nurses in different zones may be large.

Schaus et al. [2,3] proposed a CP model that directly minimizes the standard deviation of the nurses' workloads using the SPREAD constraint [1]. Results showed significant improvements in both solution quality and computational efficiency compared to the MILP model. The CP approach is able to solve problems with two zones exactly, but not very efficiently without using an approximation of the number of nurses assigned to each zone and further decomposition. While the approximation and decomposition techniques solve the problem quickly, optimality is not guaranteed. In this paper, we are interested in exact solutions.

3 Mathematical Models

Formally, the load balancing nurse-to-patient assignment problem is defined by a finite set of m patients, a finite set of n nurses, and a finite set of p zones. In addition, let P_k denote the set of patients in zone k, thus $\{P_1, \ldots, P_p\}$ forms a partition of $\{1, \ldots, m\}$. For each patient, i, a non-negative integer, A_i, represents his/her acuity. The mean of the nurses' workload is therefore computed as $\mu = \sum_{i=1}^{m} A_i/n$. The sum of the acuity levels of the patients assigned to a nurse cannot exceed $MaxAcuity$ and the total number of assigned patients cannot exceed $MaxPatients$. Each patient can only be assigned to one nurse, and each nurse can only be assigned to one zone. The objective is to minimize the variance or, equivalently the standard deviation, of the nurses' workload, measured as the total acuity assigned to the nurses.

3.1 The CP Model

In the state-of-the-art exact CP model [2], the decision variables are defined as follows:

- N_i denotes the nurse assigned to patient i.
- W_j denotes nurse j's workload.
- σ denotes the standard deviation of the variables $W_1, \ldots, W_n$.

The CP model is shown in Fig. 1. Constraint (2) is the SPREAD constraint, which relates a set of variables to their mean and standard deviation. Constraint (3) is a global packing constraint that governs the packing of items, corresponding to patients with "size" equal to their acuity levels, into bins corresponding to the workload of each nurse. Constraint (4) is the GCC placing the

$$
\begin{aligned}
\min \quad & \sigma && (1)\\
\text{s.t.} \quad & \texttt{spread}(\{W_1,\ldots,W_n\},\mu,\sigma), && (2)\\
& \texttt{multiknapsack}(\{N_1,\ldots,N_m\},\{A_1,...,A_m\},\{W_1,\ldots,W_n\}), && (3)\\
& \texttt{cardinality}(\{N_1,\ldots,N_m\},\{1,\ldots,n\},\{1,\ldots,MaxPatients\}), && (4)\\
& \texttt{pairwiseDisjoint}(\{Z_1,\ldots,Z_p\}), && (5)\\
& W_j \in \{min\{A_i\},\ldots,MaxAcuity\}, \qquad j=1,\ldots,n && (6)\\
& N_i \in \{1,\ldots,n\}. \qquad i=1,\ldots,m && (7)
\end{aligned}
$$

Fig. 1: The CP model of Schaus et al. [2].

limit on each nurse in terms of number of assigned patients. Constraint (5) expresses that a nurse can only work in one zone during the shift, where Z_k is the set of nurses assigned to zone k, i.e., $Z_k = \bigcup_{i \in P_k} N_i$. The `pairwiseDisjoint` constraint enforces pairwise empty intersections among variables representing the set of nurses working in each zone. Constraint (6) expresses bounds on the workload of each nurse. Since there are always more patients than nurses, each nurse will be assigned to at least one patient and, therefore, the W_j variables have a lower bound equal to the minimum acuity among all patients.

A customized search heuristic is an important aspect of the success of the CP model. First, the symmetry arising from the interchangeability of the nurses is dynamically broken during search by exploiting the equivalence among all nurses who have not yet been assigned a patient. Second, problem-specific variable and value ordering heuristics are implemented: the unassigned patient with the highest acuity is selected and assigned to the nurse with the current smallest workload.

3.2 The MIQP Model

We propose a mixed integer quadratic programming (MIQP) model for the problem that is mathematically equivalent to the CP model. In addition to μ, σ, and the W_j variables, we define two additional decision variables as follows:

- x_{ij} is equal to 1 if patient i is assigned to nurse j.
- z_{jk} is equal to 1 if nurse j is assigned to work in zone k.

The MIQP model is given in Fig. 2. The objective function is stated in (8). Constraint (9) specifies the quadratic relationship between the standard deviation, σ, and the workload variables, W_j. Constraint (10) ensures that each patient is assigned to exactly one nurse. Constraint (11) ensures that each nurse is assigned to exactly one zone. Constraint (12) calculates the workload of each nurse by summing over all patients. Constraint (13) ensures that the maximum number of patients assigned per nurse does not exceed the capacity of the nurse and that a nurse is only assigned patients from his/her assigned zone. Constraint (14) is the symmetry breaking constraint.

$$
\begin{aligned}
\min\quad & \sigma && && (8)\\
\text{s.t.}\quad & \sigma \geq \sqrt{\frac{\sum_{j=1}^{n}(W_j-\mu)^2}{n}}, && && (9)\\
& \sum_{j=1}^{n} x_{ij} = 1, && i = 1,\ldots,m && (10)\\
& \sum_{k=1}^{p} z_{jk} = 1, && j = 1,\ldots,n && (11)\\
& W_j = \sum_{i=1}^{m} x_{ij}A_i, && j = 1,\ldots,n && (12)\\
& \sum_{i\in P_k} x_{ij} \leq z_{jk} MaxPatients, && j = 1,\ldots,n, \quad k = 1,\ldots,p && (13)\\
& W_j \leq W_{j+1}, && j = 1,\ldots,n-1 && (14)\\
& x_{ij} \in \{0,1\}, && i = 1,\ldots,m, \quad j = 1,\ldots,n && (15)\\
& z_{jk} \in \{0,1\}, && j = 1,\ldots,n, \quad k = 1,\ldots,p && (16)\\
& W_j \in [min\{A_i\}, MaxAcuity]. && j = 1,\ldots,n && (17)
\end{aligned}
$$

Fig. 2: The MIQP model.

3.3 The CIP Model

The CIP model adds global constraints to the MIQP model. To do this, we include the N_i variable with the same meaning as in the CP model: the index of the nurse assigned to patient i. We link N_i to x_{ij} as follows:

$$N_i = \sum_{j=1}^{n} x_{ij} j. \qquad i = 1,\ldots,m \qquad (18)$$

The global constraints added to the MIQP model to form the CIP model are Constraints (2) and (4) from the CP model (Fig. 1). The complete CIP model is the MIQP model in Fig. 2 plus Constraints (18), (2), and (4).

Note that there is no need for an explicit QUADRATIC global constraint in the CIP model. The SCIP solver used for the CIP models recognizes the quadratic nature of Constraint (9) and automatically includes the QUADRATIC constraint.

4 Global Constraints

Three global constraints are included in the CIP model: QUADRATIC, SPREAD, and GCC. For each constraint, bounds propagation is used as the underlying representation in the solver used, SCIP, employs an interval representation of variable domains. As the functionality of global constraints in CIP is more general than in CP, we discuss each constraint in this section.

4.1 The Quadratic Constraint

The QUADRATIC constraint [8] is used to reason about constraints with quadratic terms and consists of a set of n variables $\{W_1, \ldots, W_n\}$, a $n \times n$ symmetric matrix $\boldsymbol{A}$, and n-dimensional vectors $\boldsymbol{b}$, $\boldsymbol{l}$ and $\boldsymbol{u}$. The representation is given as follows:

$$\texttt{quad}(\{W_1, \ldots, W_n\}, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{l}, \boldsymbol{u}),$$

where $\boldsymbol{A} \in \mathbb{Q}^{n \times n}$, $\boldsymbol{b} \in \mathbb{Q}^n$, $\boldsymbol{l} \in \mathbb{Q}^n$ and $\boldsymbol{u} \in \mathbb{Q}^n$. The constraint ensures the following condition:

$$\boldsymbol{l} \leq \sum_{i=1}^{n} \sum_{j=1}^{n} A_{i,j} W_i W_j + \sum_{i=1}^{n} b_i \leq \boldsymbol{u}.$$

We use the existing QUADRATIC constraint in SCIP [8] with its default parameter values. It implements a number of problem solving techniques including bounds propagation, addition of linear relaxations, cutting plane generation, problem reformulation, and primal heuristics.

4.2 The Global Cardinality Constraint

The GCC constraint consists of a set of m variables $\{N_1, \ldots, N_m\}$, a set of n values $\{v_1, \ldots, v_n\}$, and a set of n pairs of values $[l_j, u_j]$, for each v_j. The constraint is satisfied if and only if each v_j is assigned at least l_j times and at most u_j times to the N_i variables. The representation is given as follows:

$$\texttt{cardinality}(\{N_1, ..., N_m\}, \{v_1, ..., v_n\}, \{[l_1, u_1], ..., [l_1, u_n]\}).$$

We use the bound consistency filtering algorithm due to Quimper et al. [12].

Relaxation Linear relaxations have a central role in mixed integer programming. Given an MILP, which is, in general, NP-hard, the standard relaxation arises from ignoring the integrality constraints on the integer variables resulting in a polytime solvable linear program (LP). The LP plays numerous roles in search including providing a dual bound on a problem that is compared to the current best solution to prune search sub-trees in which no improving solution exists and in providing a basis for search heuristics.

A substantial body of work has developed linear relaxations for global constraints (e.g., [13,14,15]). We implement an existing relaxation based on the MILP representation of GCC [15]. Using the notation for GCC introduced above, if we define an additional binary variable $x_{ij} = 1$ if $N_i = v_j$, an exact formulation of GCC is presented in Fig. 3. This formulation is used by the solver to form a

$$\sum_{j=1}^{n} x_{ij} = 1, \qquad i = 1, \ldots, m \tag{19}$$

$$l_j \leq \sum_{i=1}^{m} x_{ij} \leq u_j, \qquad j = 1, \ldots, n \tag{20}$$

$$x_{ij} = 0, \qquad \forall i, j \notin D_{x_i} \tag{21}$$

$$x_{ij} \in \{0, 1\}, \qquad \forall i, j \tag{22}$$

$$N_i = \sum_{j=1}^{n} v_j x_{ij}. \qquad i = 1, \ldots, m \tag{23}$$

Fig. 3: A MILP formulation of GCC [15].

linear relaxation of the GCC constraint by ignoring the integrality constraint on x_{ij} (i.e., Constraint (22)) and replacing it with $x_{ij} \in [0, 1]$.

Our CIP model already contains linear constraints that are identical to some of those in Fig. 3. Specifically, Constraint (10), (15), and (18) in the CIP model are identical Constraints (19), (22), and (23), respectively. As Constraint (21) does not fix any x_{ij} variables in our problem, the only constraint we add to the CIP model is Constraint (20).

Cutting Planes With the importance of the linear relaxation to MILP solving, techniques for improving it via the addition of cutting planes (i.e., implied linear constraints) have been developed [16]. We use the cutting planes for GCC due to Hooker [15].

For the CIP model we generate one inequality constraint using all x_i variables:

$$\sum_{i=1}^{n} p_i v_i \leq \sum_{j=1}^{m} N_j \leq \sum_{i=1}^{n} q_i v_i, \tag{24}$$

where

$$p_i = \min\Big\{u_i\ m - \sum_{j=1}^{i-1} p_j - \sum_{j=i+1}^{n} l_j\Big\}, \quad i = 1, \ldots, n \tag{25}$$

$$q_i = \min\Big\{u_i\ m - \sum_{j=i+1}^{n} q_j - \sum_{j=1}^{i-1} l_j\Big\}. \quad i = n, \ldots, 1. \tag{26}$$

Note that p_i and q_i are the maximum number of times that v_i can be selected when, respectively, minimizing and maximizing the sum of the x_i variables, assuming, without loss of generality, that the v_is are ordered from smallest to largest. Similar inequalities including subsets of the x_i variables ranging in cardinality from 2 to $m - 1$ are also valid and could result in a smaller search space. However, our preliminary results indicated that the large number of the constraints added to the problem results in a very large model, reducing search efficiency. Please see Hooker [15] for more details.

4.3 The Spread Constraint

The SPREAD constraint was first proposed by Pesant [1] to achieve the balancing of a set of values based on statistical concepts. A bound consistency algorithm was proposed in the same paper and a simplified and extended filtering algorithm was presented in Schaus et al. [9].

The SPREAD constraint enforces a given mean μ and maximum standard deviation σ among a set of n variables $\{W_1, \ldots, W_n\}$. It can be defined as follows:

$$\texttt{spread}(\{W_1, \ldots, W_n\}, \mu, \sigma).$$

The filtering algorithm reduces the domains of the W_j variables based on μ and σ in $O(n^2)$ time-complexity. Another algorithm filters values in the domain of σ based on domains of the variables W_j in quadratic time [9]. Both algorithms are implemented in the SPREAD constraint in this paper. The complete SPREAD constraint also propagates from W_j and σ to μ. However, since in our problem the total acuity load and number of nurses are fixed and known, the mean, μ, is always fixed to a single value.

Relaxation A simple relaxation of the SPREAD constraint is a single linear equality that enforces the mean among all the variables in the SPREAD constraint. The constraint guarantees that the sum of all variables W_j is equal to $\mu \times n$, where n is the number of variables.

$$\sum_{j=1}^{n} W_j = \mu \times n. \tag{27}$$

Cutting Planes When the objective is to minimize the standard deviation, the quadratic objective function can be formulated as follows:

$$\min \sigma = \sqrt{\frac{\sum_{j=1}^{m} (W_j - \mu)^2}{n}}, \tag{28}$$

which can be re-written in the following form:

$$\min \|\boldsymbol{\mu} - \boldsymbol{I}\boldsymbol{W}\|_2^2, \tag{29}$$

where $\boldsymbol{I} \in \mathbb{Z}^{n \times n}$ is the identity matrix, $\boldsymbol{\mu} = \mu \boldsymbol{e}$, $\boldsymbol{\mu} \in \mathbb{R}^n$ and $\boldsymbol{W} \in \mathbb{R}^n$ are n-dimensional vectors.

Suppose the optimal integer solution $\boldsymbol{W}^*$ to the above problem satisfies the following bound

$$\|\boldsymbol{\mu} - \boldsymbol{I}\boldsymbol{W}^*\|_2^2 < \beta, \tag{30}$$

where β is a constant. Geometrically, Equation (30) is a hyper-ellipsoid with center $\boldsymbol{\mu}$.

Chang & Golub [17] proposed an efficient way to compute the smallest hyper-rectangle whose edges are parallel to the axes of the coordinate system and that includes the hyper-ellipsoid. Let $\mathcal{B}$ be the smallest hyper-rectangle including a hyper-ellipsoid of the general form $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 \leq \beta$, the lower bound $\boldsymbol{l}$ and the upper bound $\boldsymbol{u}$ can be computed as follows:

$$u_j = \left\lfloor \sqrt{\beta} \left\| \boldsymbol{A}^{-T} \boldsymbol{e}_j \right\|_2 + \boldsymbol{e}_j^T \boldsymbol{A}^{-1} \boldsymbol{y} \right\rfloor,$$

$$l_j = \left\lceil -\sqrt{\beta} \left\| \boldsymbol{A}^{-T} \boldsymbol{e}_j \right\|_2 + \boldsymbol{e}_j^T \boldsymbol{A}^{-1} \boldsymbol{y} \right\rceil.$$

Since in our problem, $\boldsymbol{A} = \boldsymbol{I}$ and $\boldsymbol{y} = \boldsymbol{\mu}$, Equation (30) is actually a hyper-sphere with center $\boldsymbol{\mu}$ with the same lower bounds and upper bounds for all the variables. Thus, the computation of the lower bounds and the upper bounds can be simplified as follows:

$$u_j = \left\lfloor \sqrt{\beta} + \mu \right\rfloor, \tag{31}$$

$$l_j = \left\lceil -\sqrt{\beta} + \mu \right\rceil. \tag{32}$$

An upper bound of the standard deviation σ_{max} can be used to compute β through the following relationship, based on Equation (28) and (29):

$$\beta = \sqrt{n\sigma_{max}^2}. \tag{33}$$

Therefore, the cutting planes $W_j \geq l_j$ and $W_j \leq u_j$ can be used to update the bounds on the W_j variables. In our implementation, these constraints are computed every time the upper bound of σ is improved. Since the constraints are globally feasible, the global bounds of the variables are updated if the new bounds are tighter than the current global bounds of the variables. We have previously applied a similar approach to solving binary quadratic programming problems within CIP [18].

We also place an initial lower bound on σ. Considering only the standard deviation minimization aspect of the original problem, the resulting relaxed problem can be defined as follows:

$$\min \quad \sigma_{relaxed}, \tag{34}$$

$$\text{s.t.} \quad \sigma_{relaxed} \geq \sqrt{\frac{\sum_{j=1}^{n}(W_j - \mu)^2}{n}}, \tag{35}$$

$$\sum_{j=1}^{n} W_j = n\mu, \tag{36}$$

$$W_j \in \mathbb{Z}, \qquad j = 1, \ldots, n, \tag{37}$$

where the optimal objective value $\sigma_{relaxed}$ is therefore a lower bound on σ, i.e., $\sigma \geq \sigma_{relaxed}$. The optimal solution to this relaxed problem can be obtained trivially by assigning W_js to either $\lceil \mu \rceil$ or $\lfloor \mu \rfloor$ that leads to the smallest standard deviation, while satisfying Constraint (36) and (37). For example, one can start with assigning all W_js to $\lceil \mu \rfloor$, and then assign the last $\lceil n(|\mu - \lceil \mu \rfloor|) \rfloor$ W_js to $\lceil \mu \rfloor + 1$ if $\lceil \mu \rfloor - \mu < 0$ ($\lceil \mu \rfloor - 1$ if $\lceil \mu \rfloor - \mu > 0$).

5 Branching Heuristics

It is well recognized in CP and MILP that the use of search heuristics can have substantial impact on problem solving performance [19,20,10,21]. One simple way to influence search without implementing new heuristics is to modify the heuristic priority of variables. In SCIP, the branching priority of variables can be modified, allowing problem-specific knowledge to be incorporated into the default heuristics. Increasing the branching priority of a set of variables means that they will be branched on earlier in the search.

We investigate two manipulations: 1) increasing the priority of the z_{jk} variables that assign nurses to zones and 2) increasing the branching priority of *both* the z_{jk} variables and the workload variables, W_j. In both conditions, we assign maximum priority to all corresponding variables.

Increasing z_{jk} Priority. We choose to increase the branching priority of the z_{jk} variables based on the intuition that they have a significant effect on many other variables. Pryor & Chinneck [21] have shown that to quickly find a feasible solution in MILP, it is often desirable to branch on variables whose assignment results in substantial change to the linear relaxation. When a z_{jk} variable is set to 1, Constraint (11) immediately results in the $p-1$ other z_{ik} variables with $i = j$ being assigned to 0. Furthermore, through Constraint (13), fixing a z_{jk} variable to 0 leads to the assignment of m x_{ij} variables to 0. Therefore, whether branching up or down on the z_{jk} variables, we expect to see a substantial change in the linear relaxation.

Increasing W_j Priority. We choose to also increase the branching priority of the W_j variables due to their expected impact on the dual bound. A second intuition for heuristics in a MILP is to branch to quickly increase the dual bound (i.e., the lower bound in a minimization problem). It is often observed that a considerable amount of the effort in solving a problem is not in finding a solution but in proving its optimality. As the primary method for such a proof is through pruning the sub-trees when the dual bound meets or exceeds the incumbent solution value, it is often useful to base branching heuristics on increasing the dual bound (e.g., [4]).

Given Constraint (9), branching on variables other than W_j will tend to lead to relaxed W_j values that are close μ, resulting in a dual bound that is close to 0. Branching on the W_j variables, in contrast, can more quickly increase the dual bound as the upper and lower bounds on other W_j variables must be changed due to Constraint (27).

Our initial experiments showed that only increasing the branching priority of the W_j variables without the z_{jk} led to poor performance due to difficulty in finding feasible solutions. We will return to this point in Section 7.

6 Computational Results

We compare the performance of three solvers and the following 11 solver-model-heuristic combinations.

- The CP model (Fig. 1) is implemented in COMET. We indicate this model by CP.
- The MIQP model (Fig. 2) is implemented in CPLEX using default parameters. This model is referred to as $MIQP$.
- The basic CIP is declaratively identical to $MIQP$ but, when solved using default SCIP, incorporates the QUADRATIC constraint. We experiment with three models corresponding to the branching priorities: CIP (default), CIP_z (increased z_{jk} priority), and $CIP_{z,w}$ (increased priority for z_{jk} and W_j).
- The CIP model augmented to include only the constraint propagation of the GCC and SPREAD constraints is indicated by a superscript p (for propagation). There are, again, three models: CIP^p, CIP^p_z, and $CIP^p_{z,w}$ corresponding to the branching priorities.
- The full CIP model, indicated by superscript f includes constraint propagation, relaxation, and cutting planes of the QUADRATIC, GCC, and SPREAD constraints: CIP^f, CIP^f_z, and $CIP^f_{z,w}$.

6.1 Experimental setup

All experiments were performed on a Intel(R) Xeon(R) CPU E5-1650 v2 3.50GHz machine (in 64 bit mode) with 16GB memory running MAC OS X 10.9.2 with one thread. The software is CPLEX v12.5, SCIP v3.0.2, and Comet v2.1-1. The CPU time limit for each run on each problem instance is 7200 seconds.

6.2 Test sets

Following the methodology of Schaus et al. [2], we generated 24 problem instances to closely resemble the original real world instances [11]. The maximum acuity for a nurse is set to $MaxAcuity = 105$ and the maximum number of patients per nurse is $MaxPatients = 3$. We generate problem instances with number of nurses, $n \in [9, 12]$, number of patients, $m \in [21, 30]$, and number of zones, $p = 2$.

6.3 Results

An overview of the results is given in Table 1. We report the arithmetic mean CPU time "arith", and the shifted geometric mean CPU time "geo" on the 24 instances.[1] The arithmetic mean on the number nodes "Nodes" (number of

[1] The shifted geometric mean time is computed as follows: $\prod(t_i + s)^{1/n} - s$, where t_i is the actual CPU time, n is the number of instances, and s is chosen as 10. Using geometric mean can decrease the influence of outliers [4].

Solver	Model	Time to opt		Nodes or Bts	Opt gap (%)	# opt	# opt found
		arith	geo				
COMET	CP	749.87	137.10	42292088	0^1	23	23
CPLEX	MIQP	5649.15	4247.26	9587569	31.0	7	18
SCIP	CIP	1877.63	565.02	4096764	1.0	20	23
	CIP^p	2795.80	746.89	4683757	3.0	15	18
	CIP^f	2193.95	542.27	4040087	7.0	19	20
	CIP_z	1605.64	267.11	3694590	19.0	20	23
	CIP_z^p	1723.08	331.85	3221449	0	20	24
	CIP_z^f	1459.78	267.16	2549701	6.0	21	21
	$\text{CIP}_{z,w}$	100.74	32.26	210485	0	24	24
	$\text{CIP}_{z,w}^p$	94.49	32.29	132342	0	24	24
	$\text{CIP}_{z,w}^f$	74.05	27.90	130012	0	24	24

Table 1: Comparison of CP, MIQP, and 9 different CIP variations. For CP, the superscript indicates the number of instances for which no feasible solution was found. The optimality gap "Opt gap" is computed only for the problem instances where feasible solutions are found.

backtracks "Bts" for CP) and the optimality gap "Opt gap" are also presented. We finally report the number of instances for which an optimal solution is found and proved "# opt" and the number of optimal solutions found "# opt found" without necessarily being proved.

The results of the MIQP model demonstrate that while CPLEX is able to find feasible solutions to all problems, it can only find and prove optimality in 7 of 24 with a substantially larger run-time as compared to the CP model.

The first set of CIP models (CIP, CIP^p, and CIP^f) show a significant gain compared to MIQP: more than twice as many problems solved to proved optimality with about half the number of nodes. We attribute this strong performance to the QUADRATIC constraint. The inclusion of constraint propagation in the GCC and SPREAD constraints *degrades* performance (CIP^p vs. CIP) both in terms of run-time and search tree size. The 15% larger search tree in particular is interesting and deserves more study. We speculate, given results below, that we may be observing a negative interaction between the constraint propagation and relaxation-based MILP-style search, indicating that we cannot necessarily expect improved performance from simply adding global constraint propagation to a MILP-style search. Comparing CIP^f to CIP shows similar run-times and tree sizes but worse solution quality for the full model. We believe that the gains from the global constraints are not large enough to out-weigh the increased computation they incur.

In the second set of CIP results, with increased branching priority for the z_{jk} variables, we see a substantial performance improvement in all models compared to the default heuristic. The inclusion of global constraints, whether just the propagation or the full model, results in smaller search trees by about 13% for

CIP_z^p and 30% for $CIP_{z,w}^f$ and better solution quality. However, the run-times are either about the same or are actually worse than the CIP_z model.

Finally, the most substantial gains arise from increasing the branching priority of both the z_{jk} variables and the W_j variables. About an order of magnitude improvement is seen compared to the previous CIP models. The inclusion of global constraint propagation and propagation plus relaxation and cutting planes leads to clear gains with search tree sizes of almost 40% smaller than $CIP_{z,w}$. Furthermore, the three CIP models solve all problems to proved optimality from 7 to 10 times faster (in arithmetic mean) than the CP model. We believe these are the first models of any type that have been able to improve on the performance of the CP state of the art.

7 Discussion

Our experimental results have demonstrated that the hybridization of CP and MIQP techniques within the framework of constraint integer programming results in a new state of the art for the load balanced nurse-to-patient assignment problem.

Primal Bounds and Dual Bounds. Analysis of the solving behavior of the MIQP and default CIP model points to the importance of the dual bound in achieving strong performance. The MIQP approach using CPLEX is able to find high quality solutions early in the search but then only improves the dual bound very slowly, often timing-out without proving optimality. The MIQP model is able to find the optimal solution for 18 of the 24 problems but can only prove optimality in 7. The primary advantage of the default CIP model over MIQP appears to be due to its rapid improvements in the dual bound.

A more complicated pattern emerges from the comparison of the full CIP model with and without the branching heuristics (CIP^f vs. CIP$_z^f$ vs. $CIP_{z,w}^f$). Fig. 4 plots the evolution of the mean primal and dual bounds over time for the three CIP models. For each instance, the primal, p, and dual, d, gaps are computed as follows: $p = (UB(\sigma) - \sigma^*)/\sigma^*$ and $d = (LB(\sigma) - \sigma^*)/\sigma^*$, where σ^* is the known optimal solution cost and $UB(\sigma)$ and $LB(\sigma)$ respectively represent the best upper and lower bounds at a given point in the search.

Consistent with our intuitions in Section 5, $CIP_{z,w}^f$ delivers tight primal and dual bounds, though it is more clearly dominant in the latter. However, the results of CIP_z^f contradict our expectations that increasing the priority of the z_{jk} variables alone would lead to strong primal bounds. We see the opposite, as CIP_z^f dominates CIP^f in terms of the dual bound while performing much worse on the primal bound. Interestingly, experiments that only increasing the priority of the W_j variables (not included here) do match our intuitions: the model performs poorly, timing out without finding *feasible* solutions to 5 problem instances. More detailed experimentation is needed to understand the reasons behind the impact of the search heuristics and, in particular, why $CIP_{z,w}^f$ performs so well.

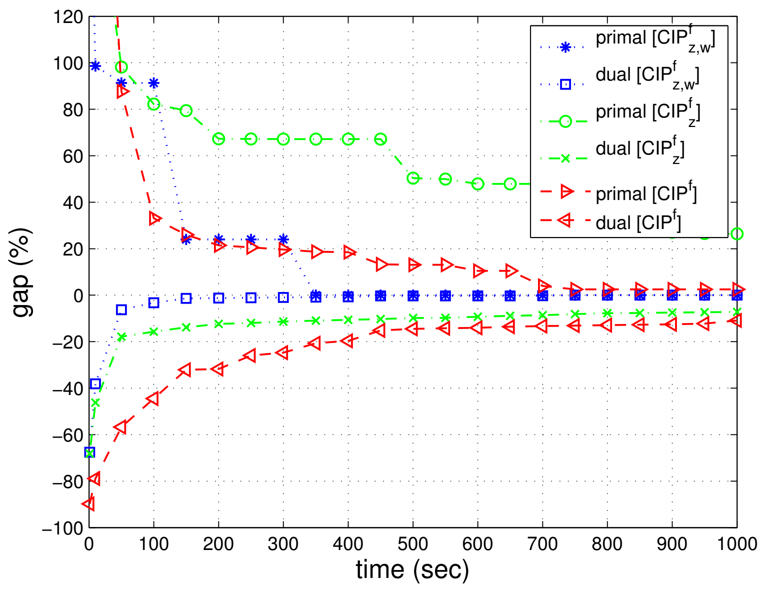

Fig. 4: Comparison of the primal and dual gaps of CIP^f, CIP^f_z and $CIP^f_{z,w}$.

The Impact of Global Constraints. Global constraints play a primary role in CP, forming the central object in both modeling and solving. Building on this role and the substantial work over the past 15 years on the hybridization of CP and MIP solving techniques [22], we are interested in exploring the concept of a global constraint as a richer object in the search process, exploiting its structure to do more than domain pruning [23]. Several works have shown the success on the pursuing of this direction. For example, global constraints can provide heuristic information [24], generate SAT clauses [25] and cutting planes [26], detect independent sub-problems [27], and decompose constraints and fix or remove variables [28].

Though we have developed a new state-of-the-art hybrid model for the nurse-to-patient assignment problem and demonstrated the reduction in search effort (in both time and nodes) from the integration of constraint propagation, linear relaxation, and cutting planes within global constraints (i.e., compare CIP_z with CIP^f_z and $CIP_{z,w}$ with $CIP^f_{z,w}$), the importance of augmented global constraints to our results should not be overstated. Without the use of the branching priorities on z_{jk} and W_j variables, none of the CIP models are able to match the performance of the CP model.[2] Therefore, it appears that the primary reason for the strong performance of the new state-of-the-art is the branching priori-

[2] The CP model also uses a problem-specific heuristic and so we believe the direct comparison of it with the $CIP^f_{z,w}$ is justified.

ties and, at best, the *interaction* of the branching priorities with the augmented global constraints. Furthermore, as the comparison of the number of nodes of CIP and CIP^p indicate, the simple addition of global constraint propagation to a MILP-style search does not necessarily result in improved performance: a more nuanced understanding of the interactions is needed.

8 Conclusions

In this paper, we developed a series of mixed integer quadratic programming and constraint integer programming models for the load balanced nurse-to-patient assignment problem. This problem has been addressed with mixed integer linear programming and constraint programming, with the latter representing the state of the art.

Our approach focused on the integration of augmented global constraints into the CIP model. In addition to constraint propagation, the global constraints implemented constraint-specific linear relaxations and cutting plane generation. Building on the existing work on the QUADRATIC [8], GCC [12,15], and SPREAD [9] constraints, we introduced a linear relaxation and cutting planes for the latter. Our empirical results demonstrate that the CIP approach substantially outperforms the MIQP model, supporting the effectiveness of the extended global constraints, but still does not compete with the CP model. To facilitate the search process, we propose problem-specific branching priorities, which greatly improve the CIP models, resulting in performance about one order of magnitude faster than CP.

References

1. Pesant, G., Régin, J.C.: Spread: A balancing constraint based on statistics. In: Principles and Practice of Constraint Programming-CP 2005. Springer (2005) 460–474
2. Schaus, P., Van Hentenryck, P., Régin, J.C.: Scalable load balancing in nurse to patient assignment problems. In: Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems. Springer (2009) 248–262
3. Schaus, P., Régin, J.C.: Bound-consistent spread constraint. EURO Journal on Computational Optimization (2013) 1–24
4. Achterberg, T.: Constraint Integer Programming. PhD thesis, Technische Universität Berlin (2007)
5. Achterberg, T.: SCIP: solving constraint integer programs. Mathematical Programming Computation **1**(1) (2009) 1–41
6. Bussieck, M.R., Vigerske, S.: Minlp solver software. Wiley Encyclopedia of Operations Research and Management Science, Wiley, Chichester (2010)
7. Koch, T., Achterberg, T., Andersen, E., Bastert, O., Berthold, T., Bixby, R., Danna, E., Gamrath, G., Gleixner, A., Heinz, S., Lodi, A., Mittelmann, H., Ralphs, T., Salvagnin, D., Steffy, D., Wolter, K.: MIPLIB 2010. Mathematical Programming Computation (2011) 1–61

8. Berthold, T., Heinz, S., Vigerske, S.: Extending a CIP framework to solve MIQCPs. In: Mixed-Integer Nonlinear Programming. Volume 154 of The IMA Volumes in Mathematics and its Applications. Springer (2012) 427–445
9. Schaus, P., Deville, Y., Dupont, P., Régin, J.C.: Simplification and extension of the spread constraint. In: Third international workshop on constraint propagation and implementation. (2006) 77–91
10. Achterberg, T., Berthold, T.: Hybrid branching. In: Proceedings of Sixth International Conference of the Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems. (2009) 309–311
11. Mullinax, C., Lawley, M.: Assigning patients to nurses in neonatal intensive care. Journal of the Operational Research Society **53**(1) (2002) 25–35
12. Quimper, C.G., Van Beek, P., López-Ortiz, A., Golynski, A., Sadjad, S.B.: An efficient bounds consistency algorithm for the global cardinality constraint. In: Principles and Practice of Constraint Programming–CP 2003, Springer (2003) 600–614
13. Refalo, P.: Tight cooperation and its application in piecewise linear optimization. In: Principles and Practice of Constraint Programming CP99. Volume 1713 of Lecture Notes in Computer Science. Springer (1999) 375–389
14. Milano, M., Ottosson, G., Refalo, P., Thorsteinsson, E.S.: The role of integer programming techniques in constraint programming's global constraints. INFORMS Journal on Computing **14**(4) (2002) 387–402
15. Hooker, J.: Integrated Methods for Optimization, 2nd edition. Springer (2012)
16. Marchand, H., Martin, A., Weismantel, R., Wolsey, L.: Cutting planes in integer and mixed integer programming. Discrete Applied Mathematics **123**(1) (2002) 397–446
17. Chang, X.W., Golub, G.H.: Solving ellipsoid-constrained integer least squares problems. SIAM Journal on Matrix Analysis and Applications **31**(3) (2009) 1071–1089
18. Ku, W.Y., Beck, J.C.: Combining discrete ellipsoid-based search and branch-and-cut for binary quadratic programming problems. In: Integration of AI and OR Techniques in Constraint Programming. Springer (2014) 334–350
19. Haralick, R.M., Elliott, G.L.: Increasing tree search efficiency for constraint satisfaction problems. Artificial Intelligence **14** (1980) 263–314
20. Beck, J.C., Prosser, P., Wallace, R.J.: Trying again to fail first. In Faltings, B., Petcu, A., Fages, F., Rossi, F., eds.: Recent Advances in Constraints. Volume 3419 of Lecture Notes in Artificial Intelligence. Springer-Verlag (2005) 41–55
21. Pryor, J., Chinneck, J.W.: Faster integer-feasibility in mixed-integer linear programs by branching to force change. Computers and Operations Research **38** (2011) 1143–1152
22. Van Hentenryck, P., Milano, M., eds.: Hybrid Optimization: Ten Years of CPAIOR. Springer (2011)
23. Beck, J.C.: Modeling, global constraints, and decomposition. In: Tenth Symposium of Abstraction, Reformulation, and Approximation. (2013)
24. Pesant, G., Quimper, C.G., Zanarini, A.: Counting-based search: Branching heuristics for constraint satisfaction problems. Journal of Artificial Intelligence Research **43**(1) (2012) 173–210
25. Schutt, A., Feydy, T., Stuckey, P.J., Wallace, M.G.: Explaining the cumulative propagator. Constraints **16**(3) (2011) 250–282
26. Bergman, D., Hooker, J.N.: Graph coloring facets from all-different systems. In: Integration of AI and OR Techniques in Contraint Programming for Combinatorial Optimzation Problems. Springer (2012) 50–65

27. Heinz, S., Ku, W.Y., Beck, J.C.: Recent improvements using constraint integer programming for resource allocation and scheduling. In: Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems. Springer (2013) 12–27
28. Heinz, S., Schulz, J., Beck, J.C.: Using dual presolving reductions to reformulate cumulative constraints. Constraints **18**(2) (2013) 166–201