BIOINFORMATICS

Vol. 22 no. 14 2006, pages e174–e183
doi:10.1093/bioinformatics/btl220

Improved Pruning algorithms and Divide-and-Conquer strategies for Dead-End Elimination, with application to protein design

Ivelin Georgiev[1], Ryan H. Lilien[1,2,3] and Bruce R. Donald[1,3,4,5,*]

[1]Dartmouth Computer Science Department, [2]Dartmouth Medical School, [3]Dartmouth Center for Structural Biology and Computational Chemistry, [4]Dartmouth Department of Chemistry and [5]Dartmouth Department of Biological Sciences, Hanover, NH 03755, USA

ABSTRACT

Motivation: Structure-based protein redesign can help engineer proteins with desired novel function. Improving computational efficiency while still maintaining the accuracy of the design predictions has been a major goal for protein design algorithms. The combinatorial nature of protein design results both from allowing residue mutations and from the incorporation of protein side-chain flexibility. Under the assumption that a single conformation can model protein folding and binding, the goal of many algorithms is the identification of the Global Minimum Energy Conformation (GMEC). A dominant theorem for the identification of the GMEC is Dead-End Elimination (DEE). DEE-based algorithms have proven capable of eliminating the majority of candidate conformations, while guaranteeing that only rotamers not belonging to the GMEC are pruned. However, when the protein design process incorporates rotameric energy minimization, DEE is no longer provably-accurate. Hence, with energy minimization, the *minimized-DEE* (*MinDEE*) criterion must be used instead.

Results: In this paper, we present provably-accurate improvements to both the DEE and MinDEE criteria. We show that our novel enhancements result in a speedup of up to a factor of more than 1000 when applied in redesign for three different proteins: Gramicidin Synthetase A, plastocyanin, and protein G.

Availability: Contact authors for source code.

Contact: Bruce.R.Donald@dartmouth.edu

1 INTRODUCTION

Desired novel protein function can result from the structure-based redesign of known protein sequences. In order to expedite the design process, a number of computational approaches for making redesign predictions have been successfully applied. In many protein design algorithms, the accuracy of the protein model is improved by incorporating protein flexibility (Street and Mayo, 1999; Jin *et al*., 2003; Jaramillo *et al*., 2001; Bolon and Mayo, 2001; Looger *et al*., 2003; Lilien *et al*., 2005). In (Najmanovich *et al*., 2000), a number of bound and unbound structures are compared, and the conclusion is drawn that only a small number of residues undergo conformational change, and that the structural changes are primarily side-chains, and not backbone. Hence, many protein design algorithms start with a rigid backbone conformation and optimize the residue sequence and the side-chain placements. Side-chain flexibility is typically modeled using a discrete set of low-energy rigid conformations, called rotamers (Lovell *et al*., 2000; Ponder and Richards, 1987). A major challenge for protein design algorithms is thus the combinatorial nature of the design process, resulting both from allowing residue mutations and from the incorporation of side-chain flexibility.

Under the assumption that a single conformation can accurately model protein folding and binding, the goal of many algorithms is the identification of the Global Minimum Energy Conformation (GMEC). It has been proven that protein design for a rigid backbone and using rotamers and a pairwise energy function is NP-hard (Pierce and Winfree, 2002; Chazelle *et al*., 2004). Hence, some heuristic approaches that do not make provable guarantees about the accuracy of the results have been developed (Street and Mayo, 1999; Kuhlman and Baker, 2000; Jin *et al*., 2003; Jaramillo *et al*., 2001; Marvin and Hellinga, 2001; Desmet *et al*., 2002; Shah *et al*., 2004). In contrast to such heuristic approaches (e.g., Monte Carlo, neural network, genetic algorithm), Dead-End Elimination (DEE) (Desmet *et al*., 1992; Lasters and Desmet, 1993) is a provable and efficient deterministic algorithm that is capable of eliminating the majority of the conformations, while guaranteeing that the GMEC is not pruned.

1.1 Traditional Dead-End Elimination

The DEE criterion (Desmet *et al*., 1992) uses rotameric energy interactions to identify and prune rotamers that are provably not part of the GMEC. The total energy of a conformation can be written as

$$E_T = E_{t'} + \sum_i E(i_r) + \sum_i \sum_{j>i} E(i_r, j_s). \qquad (1)$$

Here, i_r specifies the particular rotamer identity r at residue position i; $E_t{}'$ is the template energy (the energy of the rigid portion of the molecule); $E(i_r)$ is the self-energy (the intra-residue and residue-to-template energies) of rotamer i_r; and $E(i_r, j_s)$ is the non-bonded pairwise energy between rotamers i_r and j_s. In the original DEE criterion (Desmet *et al*., 1992), a *target* rotamer i_r could be provably

*To whom correspondence should be addressed.

pruned if a *competitor* rotamer i_t is found, such that the best (lowest) possible energy among conformations containing rotamer i_r is worse (higher) than the worst possible energy among conformations containing i_t. Hence, an alternative rotamer that is energetically more favorable than i_r exists for the entire conformation space, so i_r cannot be part of the GMEC and can thus be provably pruned. Formally, the DEE condition for pruning rotamer i_r is:

$$E(i_r) + \sum_{j \neq i} \min_s E(i_r, j_s) > E(i_t) + \sum_{j \neq i} \max_s E(i_t, j_s). \qquad (2)$$

All the pairwise and self-energy terms are precomputed and a lookup is performed during the evaluation of the DEE condition. Eq. (2) is evaluated for each target rotamer i_r until either a superior competitor i_t is found and i_r can be pruned, or there are no unexamined competitors remaining, in which case i_r would not be pruned. For a protein with n residues and a maximum of q rotamers per residue, the complexity of evaluating Eq. (2) for all target rotamers is $O(q^2n^2)$.

The evaluation of Eq. (2) for all target rotamers represents a single *DEE pruning cycle*. Since rotamers that are pruned in a given cycle are not used in the evaluation of subsequent cycles, multiple repetitions of the DEE cycle can result in pruning a larger number of rotamers. Several extensions and enhancements to the original DEE criterion use more complex energy interactions and allow for additional pruning, at the cost of some additional complexity (Desmet *et al*., 1992; Lasters and Desmet, 1993; Goldstein, 1994; Gordon and Mayo, 1998; Pierce *et al*., 2000; Looger and Hellinga, 2001). Algorithms that combine several of these extensions into the DEE cycle significantly improve the pruning efficiency, thus allowing for the redesign of larger protein motifs (Gordon *et al*., 2003; Pierce *et al*., 2000). For a summary of DEE conditions, see Fig. 3(top).

The DEE pruning cycle can be repeated until the identification of the GMEC or until no more prunings are identified during a given cycle. Although DEE is a powerful algorithm, it does not guarantee a unique solution: multiple unpruned conformations may remain after pruning with DEE is exhausted. If DEE does not produce a unique conformation, the algorithm can report an unsuccessful design (Gordon *et al*., 2003; Pierce *et al*., 2000). As an alternative, the DEE pruning stage can be followed by an enumeration stage, in which the remaining conformations are examined and the GMEC is identified. In (Leach and Lemon, 1998), A^* branch-and-bound search is used after pruning with DEE to expand a conformation tree, so that conformations are extracted in order of conformational energy; the first conformation that is returned by the A^* search is the GMEC. The need to generate all unpruned conformations is thus eliminated, resulting in a combinatorial-factor reduction in the search space. However, since the enumeration stage is still exponential in nature, an efficient DEE pruning cycle is essential for making complex design problems computationally feasible.

1.2 Minimized Dead-End Elimination

Although rotamers represent low-energy side-chain conformations, the resulting discretization of the conformation space may decrease the accuracy of the underlying model (Desmet *et al*., 2002). The motivation for performing rotameric energy minimization is thus well-founded. However, when the protein design process incorporates energy minimization, DEE is no longer provably-accurate, since a pruned conformation may subsequently minimize to a lower energy than the energy of the DEE-identified GMEC. In (Georgiev *et al*., 2006), *MinDEE*, a novel generalized DEE algorithm is presented. In contrast to *traditional-DEE* (the DEE conditions described in Sec. 1.1), MinDEE guarantees that no rotamers belonging to the *minimized-GMEC* (*minGMEC*), the conformation with the lowest energy among all energy-minimized conformations, are pruned. Thus, in order to be provably-correct, MinDEE (instead of traditional-DEE) must be used for a design process that incorporates energy minimization.

In (Georgiev *et al*., 2006), it was experimentally confirmed that traditional-DEE can prune rotamers belonging to the minGMEC. For the 9-residue active site of the phenylalanine adenylation domain of the non-ribosomal peptide synthetase (NRPS) Gramicidin Synthetase A (GrsA-PheA) (PDB id: 1AMU) (Conti *et al*., 1997), traditional-DEE and MinDEE were applied in a 2-point-mutation redesign search[1] for switching the binding affinity of the protein from Phe to Leu. Traditional-DEE was shown to prune 2 of the 9 rotamers belonging to the minGMEC. Moreover, the energy of the minGMEC was approx. 5 kcal/mol lower than the energy of the *rigid-GMEC*.[2] The results in (Georgiev *et al*., 2006) thus confirm both that traditional-DEE is not provably-correct with energy minimization and that MinDEE is more capable of returning lower-energy (and hence, more stable) conformations.

The idea underlying MinDEE is analogous to the traditional-DEE approach: rotameric energy interactions are used to determine which rotamers are provably not part of the minGMEC and can be pruned. In contrast to traditional-DEE, however, since rotamers are allowed to energy-minimize, lower and upper bounds on the self- and pairwise rotamer energies must be used, instead of the rigid-energy terms $E(i_r)$ and $E(i_r, j_s)$ in Eq. (2). We will now describe the initial MinDEE criterion, closely following (Georgiev *et al*., 2006).

Without energy minimization, a rotamer stays in the same rigid conformation, independent of the rotamer identities for the remaining residues. In contrast, with energy minimization, a rotamer r at residue i may minimize from its initial conformation in order to accommodate a change from rotamer s to rotamer u at residue j. So that one rotamer does not minimize into another, rotameric movement is constrained to a voxel of conformation space. The voxel $\mathcal{V}(i_r)$ for rotamer i_r contains all conformations of residue i within $\pm\theta$ degrees around each rotamer dihedral. Similarly, the voxel for the pair of rotamers i_r and j_s is $\mathcal{V}(i_r, j_s) = \mathcal{V}(i_r) \times \mathcal{V}(j_s)$. The self-energy of a given rotamer can change as different conformations within the voxel are assumed. We can thus define the *maximum*, *minimum*, and *range* of voxel self-energies:

$$E_{\oplus}(i_r) = \max_{z \in \mathcal{V}(i_r)} E(z), \quad E_{\ominus}(i_r) = \min_{z \in \mathcal{V}(i_r)} E(z),$$
$$E_{\oslash}(i_r) = E_{\oplus}(i_r) - E_{\ominus}(i_r).$$

The *maximum*, *minimum*, and *range* of pairwise voxel energies are defined analogously (see Fig. 3). We now define the initial

[1]In a 2-point mutation search, any 2 of the 9 active site residues are allowed to mutate simultaneously.

[2]For clarity, we will henceforth call the GMEC returned by traditional-DEE, the *rigid-GMEC*.

MinDEE criterion as:

$$\begin{aligned} E_{\ominus}(i_r) &+ \sum_{j\neq i} \min_s E_{\ominus}(i_r, j_s) - \sum_{j\neq i} \max_s E_{\oslash}(j_s) \\ &- \sum_{j\neq i} \sum_{k\neq i, k>j} \max_{s,u} E_{\oslash}(j_s, k_u) > E_{\oplus}(i_t) \\ &+ \sum_{j\neq i} \max_s E_{\oplus}(i_t, j_s). \end{aligned} \tag{3}$$

If Eq. (3) holds, then there exists a competitor i_t whose worst possible conformational energy is lower than the best possible conformational energy for the target rotamer i_r. Hence, i_r cannot belong to the minGMEC and can be provably pruned (for a proof, see (Georgiev *et al.*, 2006)). Eq. (3) for MinDEE is hence the analog of Eq. (2) for traditional-DEE. The most significant difference between traditional-DEE and MinDEE is the accounting for possible energy changes during minimization, which are incorporated through the introduction of the terms $\sum_j \max_s E_{\oslash}(j_s)$ and $\sum_j \sum_k \max_{s,u} E_{\oslash}(j_s, k_u)$. Similarly to traditional-DEE, the min and max self- and pairwise energy terms are precomputed and a lookup is performed during the pruning stage. Note that the terms $\sum_j \max_s E_{\oslash}(j_s)$ and $\sum_j \sum_k \max_{s,u} E_{\oslash}(j_s, k_u)$ can also be precomputed, since they are a function only of residue i. Thus, the MinDEE criterion (Eq. 3) can be computed as efficiently as the traditional-DEE criterion (Eq. 2).

The MinDEE criterion has been shown to be applicable both to GMEC-based and ensemble-based protein design (Georgiev *et al.*, 2006). For the ensemble-based redesign, MinDEE was applied as a provable conformational-space filter in K^*, a scoring and search protein design algorithm that incorporates energy minimization (Lilien *et al.*, 2005). Combined with A^* search, the Hybrid MinDEE-K^* algorithm introduced a significant improvement in computational efficiency over the original K^* results in (Lilien *et al.*, 2005). In MinDEE/A^* (the GMEC-based algorithm), similarly to (Leach and Lemon, 1998) for traditional-DEE, MinDEE was first used to prune a large portion of the conformational space; the minGMEC was then extracted by A^* from the remaining conformations. Although MinDEE/A^* made the search for the minGMEC computationally feasible, the provable guarantees of the algorithm resulted in more conservative pruning and, hence, in slow running times (Georgiev *et al.*, 2006). The derivation of novel techniques for improved pruning efficiency that can be incorporated into MinDEE/A^* is thus essential.

1.3 Contributions of the Paper

In this paper, we present novel provable enhancements both to traditional-DEE and MinDEE, for improved pruning efficiency. When applied in protein design searches, our enhancements yield a speedup of up to a factor of more than 1000. In particular, our paper makes the following contributions:

1. *DACS*: a provably-accurate divide-and-conquer enhancement to traditional-DEE. *DACS* is shown to obtain improved pruning efficiency and much faster running times. Due to its divide-and-conquer nature, *DACS* is especially beneficial in design problems where enumeration (Sec. 1.1) must be performed. The *DACS* algorithm is also extended to incorporate energy minimization.

2. *MinBounds*: a novel provable pruning criterion that incorporates energy minimization, generalizing the Bounds technique (Gordon *et al.*, 2003) for protein design *without* energy minimization. MinBounds prunes all rotamers i_r for which the lower bound on the energy of all conformations that contain i_r is greater than a computed reference energy.

3. Analogously to the enhancements to traditional-DEE, we derive enhancements to the initial MinDEE criterion (Eq. 3) for additional pruning. The MinDEE analogs to the traditional-DEE simple and generalized Goldstein (Goldstein, 1994), conformational splitting (Pierce *et al.*, 2000), and dead-ending pairs (Desmet *et al.*, 1992; Lasters and Desmet, 1993) conditions are presented here; the simple Goldstein criterion was previously applied in (Georgiev *et al.*, 2006).

4. A more efficient and powerful version of the MinDEE/A^* algorithm (Georgiev *et al.*, 2006), incorporating MinBounds, *DACS*, and the enhancements to the initial MinDEE criterion. The new MinDEE/A^* algorithm is shown to lead to a significant improvement in pruning efficiency;

5. Application of our novel algorithms in GMEC-based searches for redesigning plastocyanin and the β1 domain of protein G, and for switching the substrate specificity of GrsA-PheA.

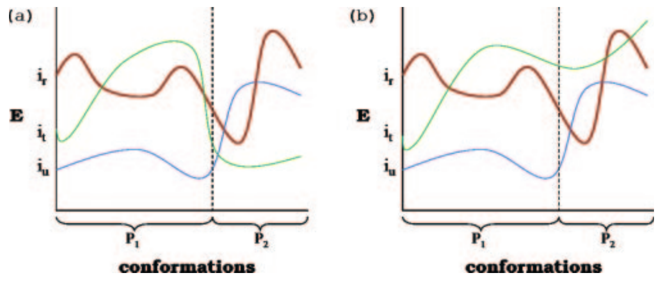

Fig. 1. Pruning with split-DEE and *DACS*. A point on the curve for rotamer i_r represents the energy of the corresponding conformation when residue i has the specific rotamer identity r. **(a)** Whereas the simple Goldstein criterion cannot prune i_r, conformational splitting can prune i_r by partitioning the conformational space. The dashed line shows a splitting of the conformational space into the two partitions P_1 and P_2. **(b)** Conformational splitting cannot prune rotamer i_r in partition P_2, so i_r must remain unpruned for the full conformational space. In contrast, *DACS* leaves i_r unpruned only for P_2; the local GMECs for P_1 and P_2 are computed and compared to obtain the overall GMEC. Note that the conformational space is discrete; continuity is shown here only for illustration purposes.

2 APPROACH

2.1 *DACS*

By partitioning the conformational search space, the original conformational splitting DEE (*split-DEE*) criterion (Pierce *et al.*, 2000) (see Fig. 3g) enhances the pruning efficiency of traditional-DEE. Fig. 1a shows a simple example of the power of conformational splitting. In Fig. 1a, the simple Goldstein criterion ((Goldstein, 1994) and Fig. 3c) would not prune rotamer i_r, since it requires that there exist a competitor rotamer with better conformational energies than i_r for *all* conformations. In contrast, when split-DEE is used, the conformational space can be divided into several partitions, such that for each partition, there is some competitor that always has better conformational energies than i_r within that partition. In Fig. 1a, the dashed line divides the space into two partitions, P_1 and P_2. With this division, the competitor rotamer

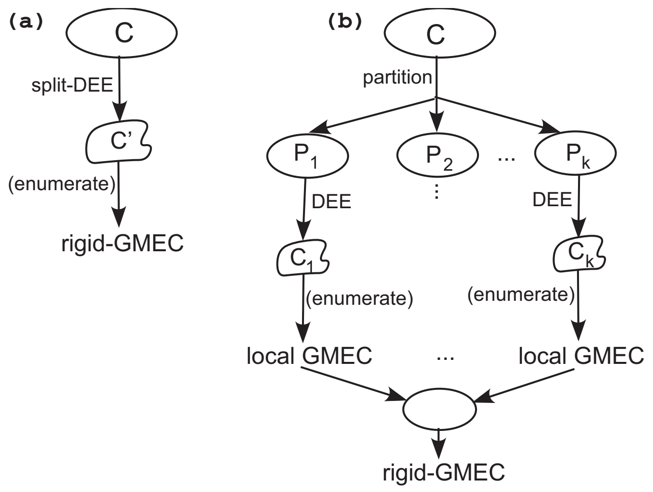

Fig. 2. Schematic of the (a) split-DEE, and (b) *DACS* algorithms. In **(a)**, the reduced set C' of conformations is obtained after split-DEE is applied to the initial conformational set C. If $|C'| = 1$, then split-DEE has output a unique solution, the rigid-GMEC; otherwise, enumeration must be performed. In **(b)**, the initial set C is first partitioned. DEE pruning is performed for each partition and the corresponding local rigid-GMEC is obtained. The lowest-energy conformation among the local GMECs for all partitions is the overall rigid-GMEC.

i_u always outperforms i_r in partition P_1, while rotamer i_t is always better than i_r in partition P_2. Thus, i_r can now be pruned, since there is always a better alternative for residue i, for any conformation. Hence, i_r is provably not part of the rigid-GMEC. The advantage of split-DEE is that no single competitor is required to outperform i_r for every conformation; as long as there exists a (different) dominant competitor for each partition, rotamer i_r can be pruned. A simplified schematic of split-DEE is given in Fig. 2a.

We now describe a modification of the split-DEE criterion that will allow for a further increase in pruning efficiency. Fig. 1b shows a different energy landscape. In this case, neither i_t nor i_u outperform i_r for all conformations in partition P_2. Thus, the original split-DEE criterion can no longer prune rotamer i_r and the potentially beneficial information that i_u is always better than i_r in partition P_1 is discarded. In general, it may be possible to prune i_r in the majority of the partitions, but so long as there exists a partition where no competitor is always better than i_r, the original split-DEE criterion must keep i_r unpruned. To remedy this loss of information, we relax the requirement that i_r be outperformed in all partitions; instead, we use a provably-accurate divide-and-conquer approach.

As in the original split-DEE criterion, we divide the conformational space into partitions. *Within* each partition, we apply DEE pruning to determine if there exists a competitor at residue i that always outperforms rotamer i_r. We then identify the *local* rigid-GMEC, restricted to the current partition, independently of the other partitions. If DEE pruning does not produce a unique solution, enumeration of the conformations in the current partition must be performed. The lowest-energy conformation among the local rigid-GMECs for all partitions is the *overall* rigid-GMEC (the rigid-GMEC among all conformations, for all partitions). We call this new approach *DACS* (**D**ivide-**A**nd-**C**onquer **S**plitting) (Fig. 2b). Note that in Fig. 1b, rotamer i_r is still unpruned in partition P_2, so the enumeration stage for P_2 must consider conformations containing i_r. However, in partition P_1, rotamer i_r can be provably pruned and hence all conformations in P_1 containing i_r can be eliminated from further consideration. With split-DEE, the conformations containing i_r for *both* partitions must still be enumerated. Hence, the general advantage of *DACS* over split-DEE is the ability to prune an additional combinatorial subset of the conformational space by exploiting partition-specific prunings.

The DEE pruning stage in *DACS* can incorporate any combination of the available provably-accurate traditional-DEE techniques (e.g., simple Goldstein and split-DEE). The enumeration stage is implemented using A^* search, which results in an additional combinatorial-factor reduction in the search space (see Sec. 1.1).

Several approaches based on ideas related to conformational splitting have been previously described. In (Looger and Hellinga, 2001), a generalized version of the split-DEE algorithm that is capable of pruning rotamer clusters, and not just single rotamers, was derived independently from (Pierce *et al*., 2000). A *split flags* technique was introduced in (Gordon *et al*., 2003) that is closely related to the approach in (Looger and Hellinga, 2001). With split flags, if a target rotamer i_r cannot be pruned for all partitions, the partitions in which i_r *can* be pruned are flagged as dead-ending. These split flags effectively represent dead-ending rotamer pairs.[3] Since the dead-ending pairs are not used in the evaluation of the DEE equations (e.g., Eq. 2), more single dead-ending rotamers may be identified in the subsequent DEE cycles.

Thus, both *DACS* and the split flags technique use pruning information that is otherwise discarded by split-DEE. However, there is one major advantage of the *DACS* algorithm over split flags, that can be attributed to the divide-and-conquer paradigm. Since the cost of expanding the A^* search tree depends combinatorially on the number of rotamers for each residue position,[4] a divide-and-conquer approach (in which the number of rotamers for each partition is reduced) can be more efficient than finding the global solution directly. Hence, for design problems in which the enumeration stage cannot be avoided, *DACS* should be especially useful.

In (Desmet *et al*., 1997), a divide-and-conquer algorithm for DEE pruning was described. In this algorithm, a list of dead-ending rotamers is constructed for each part of the divided conformational space; the *intersection* of all such lists gives the final list of pruned rotamers. Hence, this algorithm suffers from the same drawback as split-DEE: since a rotamer i_r cannot be pruned unless it is identified as dead-ending in all parts of the conformational space, potentially beneficial pruning information is often discarded.

The *DACS* algorithm benefits *both* from its divide-and-conquer nature and from the use of partition-specific prunings; *DACS* thus presents advantages over the other algorithms discussed in this section.

Correctness

We now prove the correctness of the *DACS* algorithm. Let C be the initial set of conformations and let q be the number of partitions P_i, $1 \le i \le q$, into which C is divided. Proposition 1 proves that *DACS* correctly identifies the local rigid-GMEC for each partition.

[3]In a dead-ending rotamer pair (i_r, j_s), either i_r or j_s may be part of the GMEC, but not both.

[4]For a protein with n residues and at most q rotamers per residue, the worst-case cost of expanding the A^* conformation tree is $O(q^n)$.

Proposition 2 shows that the overall GMEC is obtained as the lowest-energy conformation among the local GMECs, thus completing the proof of correctness for *DACS*.

PROPOSITION 1. *DACS identifies the local rigid-GMEC for each partition P_i.*

Proof: Let C_j denote the set of conformations for a given partition P_j, for an arbitrary j. Since the DEE pruning stage in *DACS* incorporates only provably-accurate traditional-DEE techniques the rigid-GMEC $g_j \in C_j$ is guaranteed not to be pruned. The rigid-GMEC for P_j is then extracted using the A^* search. □

PROPOSITION 2. *Let g_i be the local rigid-GMEC for partition P_i and let $E(g_i)$ be the total conformational energy of g_i. Then the overall rigid-GMEC is obtained as* $\text{argmin}_{gi}\, E\,(g_i)$, *for* $1 \leq i \leq q$:

Proof: We give a proof by contradiction. Let $h \in C$; $h \neq g_i$, $\forall i$, be the overall rigid-GMEC, so that $E(h) < \min_i E(g_i)$; that is, the overall rigid-GMEC h is not a local rigid-GMEC. By definition, h can be in exactly one partition of C; let this partition be P_j. It follows that $E(h) < E(g_j)$, so g_j is not the local rigid-GMEC for partition P_j. We thus have a contradiction. Hence, h must be a local GMEC; the lowest-energy local GMEC, $\text{argmin}_{g_i}\, E(g_i)$, for $1 \leq i \leq q$, is the overall rigid-GMEC. □

Partitioning

For each rotamer i_r, the original split-DEE (Pierce *et al*., 2000) forms partitions by choosing one or more of the protein residues as the *splitting positions* (*residues*).[5] Ideally, for n residues and s split positions, all $\binom{n-1}{s}$ possible combinations would be examined, until i_r can be pruned for all partitions in some combination. For $s >$ 2, however, the increased algorithmic complexity suggests the use of a *magic bullet* approach to splitting (Gordon and Mayo, 1998). With this approach, a single combination (a magic bullet) of split positions is chosen, based on a heuristic ranking criterion.

In the original split-DEE, different rotamers can be pruned using different combinations of splitting residues, since the pruning information is combined *before* the enumeration stage of the search for the rigid-GMEC. *DACS* uses partition-specific pruning information, so the prunings for one partition are generally not valid for a different partition (see Fig. 1b). If different rotamers are pruned using different splitting residues, the divide-and-conquer-type approach can no longer be used. Thus, the *DACS* partitions must be identical for all rotamers tested for pruning. To partition the set of conformations, we therefore choose t split residues, $1 \leq t \leq n$, *before* applying the *DACS* criterion; we will henceforth refer to these split residues as *major* split residues, in contrast with the original split-DEE splitting positions.

We use a magic-bullet-type approach for choosing the *major* split residues. Assuming preliminary DEE pruning has been performed, we can rank residues in terms of the corresponding *p-ratio* (the ratio of pruned rotamers to total number of rotamers). The top t residues with the lowest p-ratio are chosen as the *major* split positions. Intuitively, residues with a low p-ratio are less prone to pruning and should thus minimize the cost of not being able to prune rotamers at the split positions.[6] Note that the method for choosing the *major* split residues does not affect the correctness of the algorithm, but may affect its pruning efficiency, so alternative methods for choosing the *major* split positions can also be applied.

Complexity

For t *major* split residues and at most q rotamers per residue, *DACS* divides the conformational space into $O(q^t)$ partitions. The cost of running the DEE cycle for each partition is determined by the complexity of the DEE algorithms in the cycle. As noted in Sec 1.1, the cost of the initial DEE criterion (Desmet *et al*., 1992) is $O(q^2n^2)$. The simple Goldstein criterion (Goldstein, 1994) has a complexity of $O(q^3n^2)$. An implementation of the original split-DEE (Pierce *et al*., 2000) with $s = 1$ split positions has the same complexity as simple Goldstein, assuming ($q > n$). The computation of split flags is done during the split-DEE run at no additional complexity. Hence, for a DEE cycle in which the most costly algorithm used is split-DEE, the general complexity of *DACS* is $O\left(q^{2+s+t}n\binom{n-1}{s}\right)$, where $O\left(q^{2+s}n\binom{n-1}{s}\right)$ is the cost of each split-DEE run. With $t = 1$ (a single magic bullet split position) for *major* splitting and $s = 1$ split-DEE in the inner loop, *DACS* runs in $O(q^4n^2)$, which is less than the cost of $s = 2$ split-DEE, $O(q^4n^3)$. Note that since the computation of the results for each partition is independent of the other partitions, *DACS* is easily parallelizable, which further reduces the effective complexity of the algorithm.

2.2 MinDEE Extensions

As already discussed, extensions to the initial traditional-DEE criterion have resulted in improved computational efficiency (Desmet *et al*., 1992; Lasters and Desmet, 1993; Goldstein, 1994; Pierce *et al*., 2000). Analogous MinDEE extensions for additional pruning are presented in Fig. 3. For example, the conformational splitting extension to MinDEE in Fig. 3(h) is the analog of the original split-DEE extension to traditional-DEE (Fig. 3g). The *DACS* algorithm is easily extended to incorporate energy minimization; in order to only prune rotamers that are provably not part of the *minGMEC*, the traditional-DEE criteria (Fig. 3, top) in the DEE cycle of *DACS* must be discarded and their MinDEE equivalents (Fig. 3, bottom) used instead.

2.3 MinBounds

We now present a provably-accurate pruning technique that is based on rotameric minimum energy bounds. The technique, *MinBounds*, is analogous to the Bounds approach of (Gordon *et al*., 2003) for traditional-DEE. In contrast to Bounds, however, MinBounds is provably-correct with energy minimization. Similarly to (Georgiev *et al*., 2006), we define the lower bound B_{i_r} on the *minimized* energy of all conformations containing rotamer i_r as:

$$B_{i_r} = E_{t'} + E_{\ominus}(i_r) + \sum_{j \neq i} \min_{s} E_{\ominus}(j_s) + \sum_{j \neq i} \min_{s} E_{\ominus}(i_r, j_s)$$
$$+ \sum_{j \neq i} \sum_{k \neq i,\, k > j} \min_{s,\, u} E_{\ominus}(j_s, k_u).$$

Thus, B_{i_r} is the best energy that a conformation can achieve after minimization if residue i has the particular rotamer identity r. Now,

[5]A *splitting position (residue)* divides the conformational space into partitions, such that each rotamer at that residue forms a separate partition.

[6]In each partition, there is only one rotamer for each *major* split residue.

Traditional-DEE	
(a) $E(i_r) - E(i_t) + \sum_{j \neq i} \min_s E(i_r, j_s) - \sum_{j \neq i} \max_s E(i_t, j_s) > 0$	(Desmet et al., 1992)
(c) $E(i_r) - E(i_t) + \sum_{j, j \neq i} \min_s (E(i_r, j_s) - E(i_t, j_s)) > 0$	(Goldstein, 1994)
(e) $E(i_r) - \sum_{x=1,T} C_x E(i_{t_x}) + \sum_{j, j \neq i} \min_s \left(E(i_r, j_s) - \sum_{x=1,T} C_x E(i_{t_x}, j_s) \right) > 0$	(Goldstein, 1994)
(g) $E(i_r) - E(i_t) + \sum_{j, j \neq h \neq i} \left(\min_s (E(i_r, j_s) - E(i_t, j_s)) \right) + (E(i_r, h_v) - E(i_t, h_v)) > 0$	(Pierce et al., 2000)
(i) $E([i_r j_s]) - E([i_u j_v]) + \sum_{h \neq i,j} \min_t E([i_r j_s], h_t) - \sum_{h \neq i,j} \min_t E([i_u j_v], h_t) > 0$	(Desmet et al., 1992; Lasters and Desmet, 1993)
Minimized-DEE	
(b) $E_\ominus(i_r) - E_\oplus(i_t) + \sum_{j \neq i} \min_s E_\ominus(i_r, j_s) - \sum_{j \neq i} \max_s E_\oplus(i_t, j_s) - \sum_{j \neq i} \max_s E_\oslash(j_s) - \sum_{j \neq i} \sum_{k \neq i, k > j} \max_{s,u} E_\oslash(j_s, k_u) > 0$	(Georgiev et al., 2006)
(d) $E_\ominus(i_r) - E_\oplus(i_t) - \sum_{j \neq i} \max_s E_\oslash(j_s) - \sum_{j \neq i} \sum_{k \neq i, k > j} \max_{s,u} E_\oslash(j_s, k_u) + \sum_{j \neq i} \min_s (E_\ominus(i_r, j_s) - E_\oplus(i_t, j_s)) > 0$	
(f) $E_\ominus(i_r) - \sum_{x=1,T} C_x E_\oplus(i_{t_x}) - \sum_{j \neq i} \max_s E_\oslash(j_s) - \sum_{j \neq i} \sum_{k \neq i, k > i} \max_{s,u} E_\oslash(j_s, k_u) + \sum_{j \neq i} \min_s \left(E_\ominus(i_r, j_s) - \sum_{x=1,T} C_x E_\oplus(i_{t_x}, j_s) \right) > 0$	
(h) $E_\ominus(i_r) - E_\oplus(i_t) - \sum_{j \neq i} \max_s E_\oslash(j_s) - \sum_{j \neq i} \sum_{k \neq i, k > j} \max_{s,u} E_\oslash(j_s, k_u) + \sum_{j \neq i,h} \left(\min_s (E_\ominus(i_r, j_s) - E_\oplus(i_t, j_s)) \right) + (E_\ominus(i_r, h_v) - E_\oplus(i_t, h_v)) > 0;$	
(j) $E_\ominus([i_r j_s]) - E_\oplus([i_u j_v]) + \sum_{h \neq i,j} \min_t E_\ominus([i_r j_s], h_t) - \sum_{h \neq i,j} \max_t E_\oplus([i_u j_v], h_t) - \sum_{h \neq i,j} \max_t E_\oslash(h_t) - \sum_{h \neq i,j} \sum_{k \neq i,j, k > h} \max_{t,w} E_\oslash(h_t, k_w) > 0$	

Fig. 3. Dead-End Elimination Pruning Conditions. A summary of the previously-described traditional-DEE pruning conditions (top) and our newly derived minimized-DEE pruning conditions (bottom). (a) is the initial criterion for traditional-DEE (Desmet *et al*., 1992), and (b) is the generalization for minimized-DEE (Eq. 3). The *simple* (d) and *general coupled* (f) minimized-DEE pruning conditions are analogous (resp.) to the corresponding *Goldstein* pruning conditions (c, e) of traditional-DEE (Goldstein, 1994). General Goldstein (e), in traditional-DEE, compares the energy of i_r to a weighted average of the interaction energies among T candidate pruning rotamers i_{t_x}. $C_x \geq 0$ is the weight given to the energy computed using rotamer i_{t_x}. The traditional conformational splitting criterion (Pierce *et al*., 2000) and the analogous MinDEE condition are given in (g) and (h), respectively. In the minimized-DEE generalization (j) of traditional Dead-Ending Pairs (i), $E_\diamond([i_r j_s]) = E_\diamond(i_r) + E_\diamond(j_s) + E_\diamond(i_r, j_s) (i \neq j), E_\diamond([i_r, j_s]), h_t) = E_\diamond(i_r, h_t) + E_\diamond(j_s, h_t) (i, j \neq h)$ where $E_\diamond \in \{E_\ominus, E_\oplus\}$.

let E_c be the minimized energy of a given conformation and E_g be the energy of the minGMEC, so that $E_c \geq E_g$. For a given rotamer i_r, if $B_{i_r} > E_c$, then $B_{i_r} > E_g$, so i_r cannot belong to the minGMEC and can thus be provably pruned.

In (Gordon *et al*., 2003), multiple Monte Carlo searches are used throughout the design process, in order to compute lower values for E_c (called the *reference energy*), so that more rotamers could be pruned by the Bounds criterion. Alternatively, in order to reduce the computational burden, MinBounds obtains E_c by energy-minimizing the wildtype only.

The MinBounds approach is most beneficial if used in a combination with the MinDEE criteria described in Sec. 2.2. Since the MinDEE conditions are conservative, a rotamer i_r cannot be pruned unless a better alternative is found, so some rotamers with bad (high) lower energy bounds may not be pruned by MinDEE. Using MinBounds with a good reference energy guarantees that rotamers with bad lower energy bounds will be pruned, further reducing the conformational search space.

Table 1. Traditional-DEE algorithms. The name of the algorithms (*left*) is shown with the corresponding sequence of pruning criteria (*right*). Each of the pruning criteria (as well as the full DEE cycle) is repeated until no further prunings are obtained. For each target rotamer i_r, *full* split-DEE attempts pruning for all possible combinations of $\binom{n-1}{s}$ split positions. The algorithms with *DACS* use $t = 1$ *major* split positions

SD_{1f}	Bounds, simple Goldstein, full $s = 1$ split-DEE;
SD_{2f}	Bounds, simple Goldstein, full $s = 1$ split-DEE, full $s = 2$ split-DEE;
SF_{2f}	Bounds, simple Goldstein, full $s = 1$ split-DEE w/ split flags, full $s = 2$ split-DEE w/ split flags;
$DACS\text{-}SD_{1f}$	SD_{1f}, followed by $t = 1$ *DACS* with a DEE stage incorporating the set of SD_{1f} criteria;
$DACS\text{-}SD_{2f}$	SD_{2f}, followed by $t = 1$ *DACS* with a DEE stage incorporating the set of SD_{2f} criteria;
$DACS\text{-}SF_{2f}$	SF_{2f}, followed by $t = 1$ *DACS* with a DEE stage incorporating the set of SF_{2f} criteria.

Table 2. Traditional-DEE redesign for GrsA-PheA (*a*), plastocyanin (*b*), and the β1 domain of protein G (surface) (*c*). The total number of conformations for cases (*a*), (*b*), and (*c*) is 4.78×10^{15}, 2.06×10^{27}, and 2.25×10^{22}, respectively. The *Enum* values show the number of remaining conformations after pruning with the algorithm given in the corresponding column; these conformations must be considered by A^* in the enumeration stage. *Time* shows the total running time (in minutes) consumed by each algorithm for the identification of the rigid-GMEC. All experiments were performed on a *single* processor.

		SD_{1f}	SD_{2f}	SF_{2f}	$DACS\text{-}SD_{1f}$	$DACS\text{-}SD_{2f}$	$DACS\text{-}SF_{2f}$
(*a*)	Enum	4.14×10^8	2.67×10^8	2.25×10^8	1.04×10^7	1.46×10^7	3.87×10^6
	Time	46.1	34.3	25.0	2.34	3.36	2.12
(*b*)	Enum	6.78×10^{12}	4.52×10^{12}	1.86×10^{12}	5.11×10^{11}	3.84×10^{11}	6.44×10^{10}
	Time	2057.1	1192.8	207.4	769.1	534.4	55.6
(*c*)	Enum	3.7×10^{12}	1.47×10^{11}	1.6×10^{10}	3.56×10^9	2.97×10^6	2.13×10^8
	Time	*	*	4540.2	171.3	6.5	154.1

* Did not complete in 10,000 minutes.

Table 3. Partition Pruning with *DACS-SF$_{2f}$* for GrsA-PheA. The conformational space was divided into 16 partitions by splitting at residue 322 (with a *p*-ratio of 29/45 after the initial pruning with SF_{2f}). The *Enum* values show the number of remaining conformations after pruning with *DACS-SF$_{2f}$*, for each of the 16 partitions. Due to rounding, these values do not sum exactly to the corresponding total number of conformations shown in Table 2.

	1	2	3	4	5	6	7	8
Enum	5.8×10^5	2.2×10^5	1.8×10^5	2.2×10^5	7.3×10^4	3.3×10^4	3.1×10^5	0.8×10^3
	9	10	11	12	13	14	15	16
Enum	1.1×10^3	2.0×10^4	5.3×10^3	1.1×10^4	2.8×10^5	4.5×10^5	4.4×10^4	1.5×10^6

3 ALGORITHMS

3.1 Traditional-DEE

The performance advantage of *DACS* for protein design without energy minimization is evaluated in comparison to the original split-DEE and split flags. The DEE pruning stage of the benchmarking algorithms is presented in Table 1. *DACS-SD$_{1f}$*, *DACS-SD$_{2f}$*, and *DACS-SF$_{2f}$* introduce an additional complexity factor of only $O(q)$, compared to, respectively, SD_{1f}, SD_{2f}, and SF_{2f} (see Sec. 2.1, **Complexity**). For all algorithms, the pruning stage is followed by an A^*-search enumeration stage.

3.2 MinDEE

We now present an improvement of the MinDEE/A^* algorithm (Georgiev *et al*., 2006), incorporating the simple Goldstein and conformational splitting extensions to the MinDEE criterion (Sec. 2.2), MinBounds (Sec. 2.3), and *DACS* for MinDEE (Sec. 2.2). In addition, the MinDEE/A^* algorithm is adapted to allow the use of the volume filter applied in the *ensemble-based* searches of (Lilien *et al*., 2005; Georgiev *et al*., 2006). The volume filter is applied to the initial set of mutation sequences.[7] pruning over- and under-packed sequences, relative to the original sequence. For each of the remaining sequences, the MinDEE analog of the *DACS-SD$_{1f}$* algorithm (Sec. 3.1) is used to eliminate the majority of the candidate conformations. A^* search is then applied in the enumeration stage to extract the minGMEC from the set of remaining conformations. Similarly to the *DACS* algorithm, the lowest-energy conformation among the rigid-GMECs for all mutation sequences is identified as the overall rigid-GMEC. If conformations within E_w of the minGMEC energy are to be generated, the pruning criteria and the A^* search can be modified accordingly (Georgiev *et al*., 2006). The application of the enhanced pruning conditions and the use of the volume filter aim at improving the pruning capabilities and the computational efficiency of the algorithm.

[7]A mutation sequence is a particular assignment of amino acid types for each residue.

4 METHODS

Structural Model. The NRPS enzyme GrsA-PheA (PDB id: 1AMU) (Conti *et al*., 1997) is used both for the traditional-DEE and MinDEE redesigns. Similarly to (Lilien *et al*., 2005; Georgiev *et al*., 2006), the residues modeled as flexible are the 9 active site residues (D235, A236, W239, T278, I299, A301, A322, I330, C331). In addition, our structural model consists of the steric shell (the 30 residues with at least one atom within 8 Å of a residue in the active site: 186Y, 188I, 190T, 210L, 213F, 214F, 230A, 234F, 237S, 238V, 240E, 243M, 279L, 300T, 302G, 303S, 320I, 321N, 323Y, 324G, 325P, 326T, 327E, 328T, 329T, 332A, 333T, 334T, 515N, and 517K), the amino acid substrate, and the AMP cofactor. The 9 flexible residues are allowed to mutate to the set (GAVLIFYWM) of hydrophobic amino acids. Traditional-DEE experiments are also performed on plastocyanin (PDB id: 2pcy) (Garrett *et al*., 1984). Based on (Gordon *et al*., 2003), we model as flexible 18 residues in the core of plastocyanin (5, 14, 21, 27, 29, 31, 37, 38, 39, 41, 72, 74, 80, 82, 84, 92, 96, 98), allowing them to mutate to the set (AVLIFYW) of hydrophobic amino acids. Similarly to (Gordon *et al*., 2003), redesign with traditional-DEE was also performed on 14 surface residues (4, 6, 8, 13, 15, 17, 42, 44, 46, 48, 49, 51, 53, 55) of the β1 domain of protein G (PDB id: 1pga) (Gallagher *et al*., 1994). The 14 residues modeled as flexible are allowed to mutate to the set (ANQSTDE); the remaining residues (except for the N-terminus) are modeled as part of the steric shell. Further, similarly to (Shah *et al*., 2004), 1pga redesign was performed on 12 core residues (3, 5, 7, 9, 20, 26, 30, 34, 39, 41, 52, 54), allowed to mutate to (GAVLIFYWM). **Rotamer Library.** Side-chain flexibility is modeled using the Richardsons' rotamer library (Lovell *et al*., 2000). **Energy Minimization.** Conformations are energy-minimized using steepest-descent minimization and the AMBER energy function (electrostatic, vdW, and dihedral energy terms) (Weiner *et al.*, 1984; Cornell *et al.*, 1995). A voxel of $\theta = \pm 9\,^{\circ}$ is allowed around each rotamer dihedral. **Volume Filter.** *(MinDEE/A* only)* Over-/under-packed mutation sequences (by more than 30Å^3) relative to wildtype GrsA-PheA are pruned.

5 RESULTS AND DISCUSSION

Traditional-DEE. The results of applying the 6 different algorithms described in Sec. 3.1 to GrsA-PheA are shown in Table 2, Case (*a*). With $s = 1$ split-DEE (SD_{1f}), the redesign process took 46.1 minutes on a single processor, but the introduction of *DACS* (*DACS*-SD_{1f}) decreased the execution time by a factor of 20. For $s = 2$ split-DEE without and with split flags (SD_{2f} and SF_{2f}, respectively), the application of *DACS* resulted in a speedup factor of approx. 10 and 12, respectively. Thus, the minor additional complexity of the algorithms incorporating *DACS* (see Sec. 3.1) is outweighed by a significant increase in computational efficiency over the corresponding algorithms without *DACS*. Moreover, *DACS* performed better even when compared to more costly algorithms: *DACS*-SD_{1f} was a factor of 10 faster than the SF_{2f} algorithm.

A major factor for the speedup associated with the *DACS* algorithms is the corresponding increase in pruning efficiency (Table 2). By using a divide-and-conquer approach to partition the conformational space and identify partition-specific prunings, *DACS* allows for additional elimination, after pruning with the original split-DEE and split flags techniques is exhausted. Table 3 shows the *DACS*-SF_{2f} pruning results for all 16 partitions. As can be seen from Table 3, the remaining conformations after the DEE stage of *DACS* differ widely for each partition, ranging from less than 1,000 (partition 8) to approx. 1.5 million (partition 16). This variation shows that a different subset of rotamers can be pruned for each of the partitions, confirming the significance of using the *DACS* partition-specific prunings.

The improved execution times of the *DACS* redesigns can further be explained by the reduced cost of expanding the A^* search trees for each partition, resulting from the divide-and-conquer approach, as opposed to expanding the single A^* tree for the full conformational space. For example, for the SF_{2f} algorithm, A^* must simultaneously consider all of the remaining 2.25×10^8 conformations, whereas the largest partition for *DACS*-SF_{2f} has only 1.5×10^6 candidate conformations.

Table 2, Case (*b*), shows the plastocyanin redesign results for the six different algorithms used. Similarly to GrsA-PheA, the *DACS* algorithms (columns $4-6$) outperform the corresponding split-DEE/split flags algorithms in columns $1-3$, resulting in a speedup of up to a factor of 4. Unlike GrsA-PheA, however, the execution time for SF_{2f} was less than that for *DACS*-SD_{1f}, although the total number of unpruned conformations for *DACS*-SD_{1f} was smaller. We can thus conclude that the overhead of expanding separate A^* trees for each partition can be outweighed only by a significant improvement in pruning efficiency. However, in all of the redesign results presented in Table 2, the addition of the *DACS* algorithm (columns $4-6$) shows the necessary substantial increase in pruning efficiency over the *respective* algorithms (without *DACS*) in columns $1-3$. Hence, we conclude that, in general, *DACS* should be used as an enhancement, and not a substitute, to the other available DEE techniques.

Table 4. MinDEE/A^* Redesign for GrsA-PheA using MA$_{new}$ (*a*) and MA$_{simple}$ (*b*). The number of conformations remaining after the volume filter is 1.7×10^8. *Pruned* shows the number and percentage (in parentheses) of conformations pruned by the MinDEE stage of the corresponding algorithm; the number of remaining unpruned conformations is shown in *Remaining*; *Minimized* represents the number of conformations generated by A^* and energy-minimized. *Time/Seq.* is the average CPU time (in minutes) for the evaluation of a single mutation sequence.

	(*a*)	(*b*)
Pruned	1.697×10^8 (99.8%)	1.66×10^8 (97.6%)
Remaining	3.86×10^5	4.0×10^6
Minimized	9.3×10^4	9.64×10^4
Time/Seq.	16.11	16.66

The core redesign of the $\beta 1$ domain of protein G was completed within 5 minutes by all six algorithms (data not shown), which precludes a differential performance comparison for this case. However, our conclusions so far are confirmed by the (more difficult) surface redesigns of $\beta 1$ of protein G (Table 2, Case *c*). When compared to the algorithms without *DACS*, the respective *DACS* algorithms show a speedup of up to three orders of magnitude. In fact, the SD_{1f} and SD_{2f} algorithms exceeded the maximum allotted time of 10,000 minutes, so the use of *DACS* for these redesigns was essential. Moreover, similarly to Case (*a*), *DACS*-SD_{1f} performed an order of magnitude better than the more costly SF_{2f}.

Note that SF_{2f} in Case (*c*) ran 20 times slower than SF_{2f} in Case (*b*), although the number of unpruned conformations for Case (*c*) was two orders of magnitude lower. This is a direct result of the expansion mechanism of A^* and implies that, in order to generate the best conformation, a larger portion of the A^* conformation tree had to be expanded for SF_{2f} in Case (*c*) than in Case (*b*). Indeed, the A^* tree in Case (*c*) contained approx. 1.9×10^6 nodes at the time of completion, whereas the Case (*b*) tree contained only 5×10^5 nodes.

Also note the increased running time of *DACS*-SD_{2f} as compared to *DACS*-SD_{1f} (Case *a*) and *DACS*-SF_{2f} as compared to *DACS*-SD_{2f} (Case *c*). This can be explained by the choice of an inefficient *major* splitting residue. To test this hypothesis, we examined a different heuristic for choosing the *major* splitting positions, so that preference is given to lower-numbered residues.[8] With the new approach, a higher-numbered residue n_{i+k} is chosen as the *major* split position if its *p*-ratio is at least a value of α lower than the *p*-ratio of the lower-numbered residue n_i, $1 \leq i \leq n$. In our experiments, we used $\alpha = 0.15$. The new splitting approach significantly reduced the running times for most *DACS* redesigns (data not shown). *DACS*-SD_{2f} and *DACS*-SD_{1f} in Case (*a*) ran in 2.15 and 2.04 minutes, respectively. The running time of *DACS*-SD_{2f} (Case *c*) remained unchanged, whereas that of *DACS*-SF_{2f} was reduced by a factor of 44 to a total of 3.5 minutes. We can thus conclude that more sophisticated alternatives for choosing the *major* splitting positions should further improve the computational efficiency.

[8]Lower-numbered residues are at lower depths of the A^* conformation tree and are thus expanded first.

The results in this section show the additional pruning power and computational speedup of the *DACS* algorithm for traditional-DEE design, compared to the original split-DEE and split flags techniques, thus confirming the significance of this new approach.

MinDEE/A^*. Results from a 2-point mutation redesign search with energy minimization for switching the binding affinity of GrsA-PheA from Phe to Leu are shown in Table 4. Our improved version of the MinDEE/A^* algorithm[9] (Sec. 3.2), Table 4(a), is compared against the original MinDEE/A^* algorithm[10] (Georgiev *et al*., 2006), Table 4(b), which uses only the MinDEE analog of the simple Goldstein criterion. In order to fairly evaluate the effects of using the novel pruning criteria presented in this paper, the original MinDEE/A^* algorithm was also modified to incorporate the volume filter described in Sec. 3.2. For our experiments, we used a value of 6.0 for E_w (Sec. 3.2). The redesigns were performed on a cluster of 36 processors.

Only 30% of the mutation sequences passed the volume filter. The application of the MinDEE criteria in MA_{new} resulted in the elimination of (99.8%) of the remaining conformations, while the same algorithmic stage in MA_{simple} eliminated only (97.6%). The number of remaining conformations that had to be considered by A^* in the enumeration stage was consequently an order of magnitude smaller for the MA_{new} algorithm. Thus, as desired, the incorporation of the novel pruning techniques significantly enhanced the pruning capabilities of the MinDEE stage.

When considering the execution times, however, the speedup resulting from the use of the MA_{new} algorithm was not significant. The reason that the increased pruning efficiency did not lead to increased computational efficiency can be explained by the role of the MinDEE stage in the MinDEE/A^* algorithm. By pruning the majority of the possible rotamers, MinDEE reduces the cost of expanding the A^* search tree.[11] Since the number of rotamers for a *single* mutation sequence is comparatively small, the overhead of expanding the A^* tree is also smaller. Hence, for a single sequence, the execution time will be dominated mostly by the conformational energy minimization, and not by the tree expansion. Since an approximately equal number of conformations are energy-minimized by both MA_{new} and MA_{simple}, the similar execution times of both algorithms are not surprising. However, the fact that the novel advanced pruning techniques resulted in a significant increase in pruning efficiency, leads to the conclusion that the improved MinDEE/A^* algorithm will be especially useful in redesigns of larger systems[12] with energy minimization where the cost of managing the search tree dominates the computational effort.

6 CONCLUSION

In this paper, we presented novel enhancements for increased pruning efficiency, applicable in protein design problems both with and without energy minimization. The additional pruning power and the divide-and-conquer nature of the *DACS* algorithm were shown to lead to a significant computational speedup over other conformational-splitting-based algorithms, for the redesigns of GrsA-PheA, plastocyanin, and $\beta 1$ of protein G. Plastocyanin and protein G redesigns were also described in (Pierce *et al*., 2000; Gordon *et al*., 2003), using conformational splitting techniques in a combination with other advanced pruning criteria, such as dead-ending pairs. It would thus be interesting to incorporate such advanced pruning techniques into the *DACS* algorithms, in order to facilitate the faster design of larger systems. Moreover, since the choice of *major* splitting residues was shown to impact the efficiency of the algorithm, a further improvement of *DACS* could involve the derivation of a better approach for choosing the split positions. For larger systems, the use of multiple *major* split positions should also prove beneficial.

Our improved MinDEE/A^* algorithm incorporated the MinBounds technique, the simple Goldstein and split-DEE extensions to MinDEE, and the MinDEE version of *DACS*, resulting in a significant improvement in pruning efficiency over the original MinDEE/A^* algorithm. Similarly to traditional-DEE, further improvements to MinDEE/A^* could include the incorporation of $s = 2$ split-DEE and the split-flags techniques, as well as other advanced pruning criteria. As suggested by our results, in order to benefit from the increased pruning efficiency, MinDEE/A^* should be applied to larger systems, where the cost of expanding the search tree in the enumeration stage, rather than the energy minimization, will dominate the computation. MinDEE experiments on larger systems are currently under way and will be reported in future work.

The pruning techniques presented in this paper add to the power of available protein design algorithms and can be an important step towards the development of algorithms for the efficient solution of increasingly more computationally-expensive design problems. More efficient algorithms will also allow the use of improved models (e.g., larger rotamer libraries, improved energy functions, and the incorporation of backbone flexibility), thus increasing the accuracy of the design predictions.

ACKNOWLEDGEMENTS

We thank Prof. A. Anderson, Dr. S. Apaydin, Mr. J. MacMaster, Mr. A. Yan, Mr. B. Stevens, and all members of the Donald Lab for helpful discussions and comments. This work is supported by grants to B.R.D. from the National Institutes of Health (R01 GM-65982), and the National Science Foundation (EIA-0305444).

REFERENCES

Bolon,D. and Mayo,S. (2001) Enzyme-like proteins by computational design. *PNAS USA*, **98**, 14274–14279.

Chazelle,B., Kingsford,C. and Singh,M. (2004) A semidefinite programming approach to side-chain positioning with new rounding strategies. *INFORMS Journal on Computing, Computational Biology Special Issue*, **16**(4), 380–392.

Conti,E., Stachelhaus,T., Marahiel,M. and Brick,P. (1997) Structural basis for the activation of phenylalanine in the non-ribosomal biosynthesis of Gramicidin S. *EMBO J*., **16**: 4174.

Cornell,W., Cieplak,P., Bayly,C., Gould,I., Merz,K., Ferguson,D., Spellmeyer,D., Fox,T., Caldwell,J. and Kollman,P. (1995) A second generation force field for the simulation of proteins, nucleic acids and organic molecules. *JACS*, **117**, 5179–5197.

Desmet,J., Maeyer,M., Hazes,B. and Lasters,I. (1992) The dead-end elimination theorem and its use in protein side-chain positioning. *Nature*, **356**, 539–542.

Desmet,J., Maeyer,M.D. and Lasters,I. (1997) Theoretical and algorithmical optimization of the dead-end elimination theorem. In Altman,R., Dunker,A., Hunter,L. and Klein,T. (eds), *Pac Symp Biocomput*., 122–33.

[9]For convenience, we will henceforth refer to the this version as MA_{new}.
[10]Henceforth referred to as MA_{simple}.
[11]As noted before (Sec. 2.1), this cost depends combinatorially on the number of rotamers for each residue position.
[12]For example, larger proteins, a larger number of flexible residues, or the simultaneous redesign of multiple mutation sequences.

Desmet,J., Spriet,J. and Lasters,I. (2002) Fast and accurate side-chain topology and energy refinement (FASTER) as a new method for protein structure optimization. *Proteins*, **48**, 31–43.

Gallagher,T., Alexander,P., Bryan,P. and Gilliland,G.L. (1994) Two crystal structures of the B1 immunoglobulin-binding domain of streptococcal protein G and comparison with NMR. *Biochemistry*, **33**, 4721–4729.

Garrett,T.P., Clingeleffer,D.J., Guss,J.M., Rogers,S.J. and Freeman,H.C. (1984) The crystal structure of poplar apoplastocyanin at 1.8-a resolution. The geometry of the copper-binding site is created by the polypeptide. *J. Biol. Chem.*, **259**, 2822–2825.

Georgiev,I., Lilien,R. and Donald,B.R. (2006) A novel minimized dead-end elimination criterion and its application to protein redesign in a hybrid scoring and search algorithm for computing partition functions over molecular ensembles. In *Proceedings of The Tenth Annual International Conference on Research in Computational Molecular Biology (RECOMB),* pp. 530–545. Venice, Italy. Springer Berlin, RECOMB 2006, Lecture Notes in Computer Science, LNBI 3909.

Goldstein,R. (1994) Efficient rotamer elimination applied to protein side-chains and related spin glasses. *Biophys. J.*, **66**, 1335.

Gordon,D., Hom,G., Mayo,S. and Pierce,N. (2003) Exact rotamer optimization for protein design. *J Comput Chem*, **24**, 232–243.

Gordon,D. and Mayo,S. (1998) Radical performance enhancements for combinatorial optimization algorithms based on the dead-end elimination theorem. *J. Comput. Chem.*, **19**, 1505–1514.

Jaramillo,A., Wernisch,L., Héry,S. and Wodak,S. (2001) Automatic procedures for protein design. *Comb. Chem. High Throughput Screen.*, **4**, 643–659.

Jin,W., Kambara,O., Sasakawa,H., Tamura,A. and Takada,S. (2003) De novo design of foldable proteins with smooth folding funnel: Automated negative design and experimental verification. *Structure*, **11**, 581–591.

Kuhlman,B. and Baker,D. (2000) Native protein sequences are close to optimal for their structures. *PNAS*, **97**, 10383–10388.

Lasters,I. and Desmet,J. (1993) The fuzzy-end elimination theorem: correctly implementing the side chain placement algorithm based on the dead-end elimination theorem. *Protein Eng.*, **6**, 717–722.

Leach,A. and Lemon,A. (1998) Exploring the conformational space of protein side chains using dead-end elimination and the A* algorithm. *Proteins*, **33**, 227–239.

Lilien,R., Stevens,B., Anderson,A. and Donald,B.R. (2005) A novel ensemble-based scoring and search algorithm for protein redesign, and its application to modify the substrate specificity of the Gramicidin Synthetase A phenylalanine adenylation enzyme. *Journal of Computational Biology*, **12**(6–7), 740–761.

Looger,L., Dwyer,M., Smith,J. and Hellinga,H. (2003) Computational design of receptor and sensor proteins with novel functions. *Nature*, **423**, 185–190.

Looger,L. and Hellinga,H. (2001) Generalized dead-end elimination algorithms make large-scale protein side-chain structure prediction tractable: Implications for protein design and structural genomics. *J. Mol. Biol.*, **307**, 429–445.

Lovell,S., Word,J., Richardson,J. and Richardson,D. (2000) The penultimate rotamer library. *Proteins*, **40**, 389–408.

Marvin,J. and Hellinga,H. (2001) Conversion of a maltose receptor into a zinc biosensor by computational design. *PNAS*, **98**, 4955–4960.

Najmanovich,R., Kuttner,J., Sobolev,V. and Edelman,M. (2000) Side-chain flexibility in proteins upon ligand binding. *Proteins*, **39**(3), 261–8.

Pierce,N., Spriet,J., Desmet,J. and Mayo,S. (2000) Conformational splitting: a more powerful criterion for dead-end elimination. *J. Comput. Chem.*, **21**, 999–1009.

Pierce,N. and Winfree,E. (2002) Protein design is *NP*-hard. *Protein Eng.*, **15**, 779–782.

Ponder,J. and Richards,F. (1987) Tertiary templates for proteins: Use of packing criteria in the enumeration of allowed sequences for different structural classes. *J. Mol. Biol.*, **193**, 775–791.

Shah,P., Hom,G. and Mayo,S. (2004) Preprocessing of rotamers for protein design calculations. *J Comput Chem*, **25**, 1797–1800.

Street,A. and Mayo,S. (1999) Computational protein design. *Structure*, **7**, R105–R109.

Weiner,S., Kollman,P., Case,D., Singh,U., Ghio,C., Alagona,G., Profeta,S. and Weiner,P. (1984) A new force field for molecular mechanical simulation of nucleic acids and proteins. *J. Am. Chem. Soc.*, **106**, 765–784.