Conformational Splitting: A More Powerful Criterion for Dead-End Elimination

N. A. PIERCE,[1,*] J. A. SPRIET,[2] J. DESMET,[2,†] S. L. MAYO[3]

[1]*Division of Biology, California Institute of Technology, Pasadena, California*
[2]*Interdisciplinary Research Center, Katholieke University Leuven, Campus Kortrijk, Kortrijk, Belgium*
[3]*Howard Hughes Medical Institute and Division of Biology, California Institute of Technology, Pasadena, California*

Received 19 November 1999; accepted 9 March 2000

ABSTRACT: Dead-end elimination (DEE) is a powerful theorem for selecting optimal protein side-chain orientations from a large set of discrete conformations. The present work describes a new approach to dead-end elimination that effectively splits conformational space into partitions to more efficiently eliminate dead-ending rotamers. Split DEE makes it possible to complete protein design calculations that were previously intractable due to the combinatorial explosion of intermediate conformations generated during the convergence process.
 J Comput Chem 21: 999–1009, 2000

Keywords: dead-end elimination; conformational splitting; global minimum energy conformation; side-chain placement; protein design

Introduction

Protein function follows from protein structure, which in turn follows from protein sequence. Protein design efforts aim to exploit these relationships by selecting novel amino acid sequences that yield structures with enhanced stability properties or new functionality. For a given target backbone fold, the selection process consists of modeling the side-chain interactions of the various sequence alternatives and choosing the sequence that produces the most energetically favorable side chain packing.[1] Homology modeling efforts exploit these relationships in a different way by determining the unknown backbone fold of a given sequence based on similarities with related sequences of known structure. Because sequence comparisons do not establish side-chain orientations, accurate side-chain placement algorithms are also central to homology modeling endeavors.[2]

Although the side chains of a real protein adopt an equilibrium conformation out of a continuum of possible orientations, for computational purposes, it is advantageous to discretize the allowed con-

*Present address: Department of Applied Mathematics, California Institute of Technology, Pasadena, CA

†Present address: AlgoNomics NV, Kortrijk, Belgium

Correspondence to: S. L. Mayo; e-mail: steve@mayo.caltech.edu

Contract/grant sponsors: Burroughs-Wellcome Foundation; Caltech Initiative in Computational Molecular Biology; Faculty of Agricultural and Applied Biological Sciences; Belgian National Fund for Scientific Research (FWO); Howard Hughes Medical Institute

formational space into "rotamers" representing the statistically dominant side-chain orientations in naturally occurring proteins.[3–5] Theoretically, the side-chain placement problem can then be solved by evaluating the energy of each of the combinatorial alternatives represented by the rotamer library. However, for a protein with p positions and an average of n rotamers at each position, the number of possible conformations is $O(n^p)$, so that an exhaustive search of the entire combinatorial problem is infeasible.

Dead-end elimination algorithms[2] provide a deterministic approach to finding the global minimum energy conformation (GMEC) of a set of amino acid side chains anchored to specified backbone coordinates. All of the rotamers at a particular residue position are essentially in competition for inclusion in the GMEC. The idea underlying DEE algorithms is that, by comparing the energy contributions of different candidate rotamers at a given position, it is possible to identify certain rotamers which cannot exist in the GMEC. These dead-ending rotamers can be eliminated from future consideration, thus decreasing the combinatorial size of the problem.

To follow this approach, the potential function used to evaluate the conformational energy must be expressed solely in terms of pairwise interactions. The relative merits of candidate rotamers at a given position can then be ascertained without having to evaluate the total energy of all conformations using each of the candidates. Instead, only the portion of the total energy that arises from pairwise interactions with the position in question need be considered. By comparing the relative size of the pairwise energy contributions using each of the candidate rotamers at this position, it is possible to identify incompatibility with the GMEC without knowledge of the actual minimum energy. The combinatorial cost of this procedure is far less than the cost of complete enumeration of the energy of each conformation.

Starting with the original formulation of Desmet et al.[2] in the context of side-chain placement, a variety of improvements have been made to the DEE approach. In addition to considering the elimination of single rotamers, Lasters and Desmet[6] refined the use of doubles calculations to flag dead-ending rotamer pairs for more efficient elimination using the singles approach. Goldstein[7] then suggested a more powerful DEE criterion that is applicable to both singles and doubles calculations. Combining these enhancements, Lasters et al.[8] succeeded in performing side-chain placement calculations on proteins with several hundred residues.

Dahiyat and Mayo[9] extended the DEE approach to protein design by allowing rotamers from multiple amino acid types to compete at each position. The total number of rotamers is thus greatly increased for design applications, and poses a correspondingly greater demand on computational efficiency. Gordon and Mayo[10] introduced several important modifications to the Goldstein doubles approach which lead to substantial improvements in the performance of dead-end elimination for protein design. Dahiyat and Mayo[1] succeeded in redesigning a small 28-residue zinc finger using this approach. However, further algorithmic improvements will be required before significantly larger numbers of positions can be designed simultaneously using well-resolved rotamer libraries.

Although the DEE theorem greatly reduces the combinatorial complexity of the problem relative to an exhaustive search, the approach still exhibits a higher order dependence on the number of rotamers and positions. Using the Goldstein criterion, elimination of dead-ending rotamers at all positions requires the calculation of $O(n^3p^2)$ pairwise interactions. After iteratively applying singles DEE, it becomes necessary to resort to more costly doubles calculations, which require $O(n^5p^3)$ pairwise evaluations, and quickly come to dominate the computational cost of the problem.

The present work describes a new version of singles DEE that effectively splits the conformational space into partitions. A rotamer at a given position can then be eliminated if, within each of the partitions, at least one of the other candidate rotamers at that location always produces lower pairwise interaction energies. A hierarchy of splittings is defined, the simplest of which remains $O(n^3p^2)$, while substantially increasing the number of rotamers that are identified as dead-ending.

A progression of increasingly sophisticated dead-end elimination criteria will be described mathematically, as well as by a sequence of pictures that are very helpful in conveying the improvement resulting from each new approach. Complexity estimates are provided for each method and then a number of enhancements and practical considerations are discussed. Subsequently, the computational savings of the split DEE approach are demonstrated relative to existing state-of-the-art DEE methods for challenging protein design calculations.

Theoretical Foundations

Using a potential function described in terms of pairwise interactions, the total energy of a protein can be evaluated with:

$$E_{\text{total}} = E_{\text{template}} + \sum_i E(i_r) + \sum_i \sum_{j,\, j<i} E(i_r, j_u) \quad (1)$$

Here, E_{template} represents the self-energy of the backbone and $E(i_r)$ represents the energy of rotamer r at position i, including both the self-energy and the interaction energy with the backbone. The term $E(i_r, j_u)$ represents the interaction energy between rotamers r and u at positions i and j, respectively. It is precisely the calculation of this total energy for each possible conformation that DEE seeks to avoid.

ORIGINAL DEE

The original dead-end elimination criterion[2] states that a rotamer, i_r, can be eliminated if an alternative rotamer, i_t, at the same position (see Fig. 1a) satisfies:

$$E(i_r) + \sum_{j,\, j\neq i} \min_u E(i_r, j_u) > E(i_t) + \sum_{j,\, j\neq i} \max_u E(i_t, j_u) \quad (2)$$

This condition implies that i_r can be eliminated if the net energy contribution resulting from its best-case pairwise interactions with rotamers at all other positions (spanned by j_u) is still worse than that produced by the worst-case pairwise interactions of some other candidate rotamer, i_t, at the same position. To help in visualizing the significance of this criterion, Figure 2a depicts some relevant energy landscapes. Here, the abscissa represents all possible conformations of the protein and the ordinate describes the net energy contribution produced by interactions with specific rotamers at position i. It is important to note that these energy profiles are not actually continuous and are instead composed of discrete points displayed in some arbitrary ordering of conformational space. The left-hand side of eq. (2) identifies the energy corresponding to the best-case conformation A_r for rotamer i_r, and the right-hand side identifies the energies for worst-case conformations A_{t1} and A_{t2} for candidate rotamers i_{t1} and i_{t2}, respectively. Hence, in the present scenario, rotamer i_r could be eliminated by rotamer i_{t1} but not by i_{t2}.

SIMPLE GOLDSTEIN DEE ($T = 1$)

Goldstein[7] improved on this idea using the more powerful criterion:

$$E(i_r) - E(i_t) + \sum_{j,\, j\neq i} \left\{ \min_u \left[E(i_r j_u) - E(i_t j_u) \right] \right\} > 0 \quad (3)$$

which states that rotamer i_r can be eliminated if the contribution to the total energy is always reduced by using an alternative rotamer, i_t. In Figure 2b, the criterion measures the minimum difference between the profile for i_r and the profiles using other candidate rotamers. In the present example, rotamers i_{t1} and i_{t2} are both able to eliminate i_r, because the differences at the points of closest approach (B_{t1} and B_{t2}, respectively) are positive in both cases. This increased elimination power is a tremendous advantage in reducing the combinatorial size of the problem prior to resorting to doubles calculations. However, the criterion is still unable to eliminate rotamer i_r if all of the candidate i_t rotamers have energy profiles that cross the i_r profile for at least one conformation.

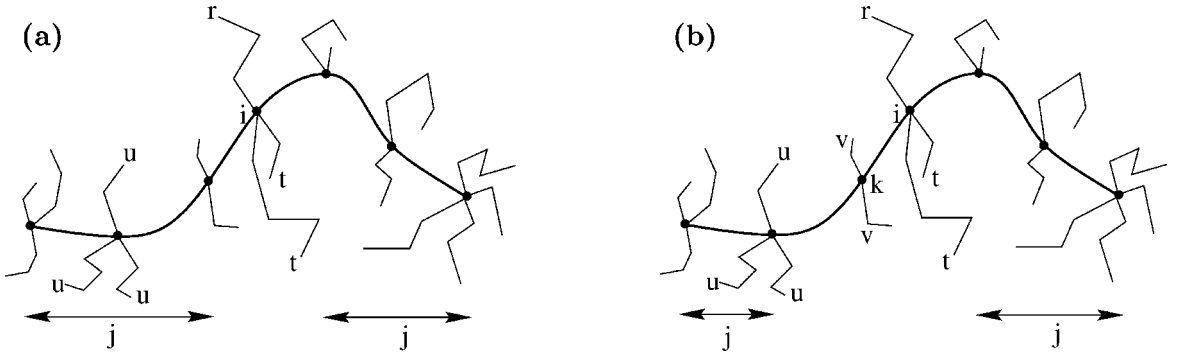

FIGURE 1. Rotamer notation for dead-end elimination. The thick line represents the protein backbone, the filled circles indicate residue positions, and the thin lines emanating from each residue are side-chain rotamers. (a) Original and Goldstein DEE. (b) Simple split DEE.

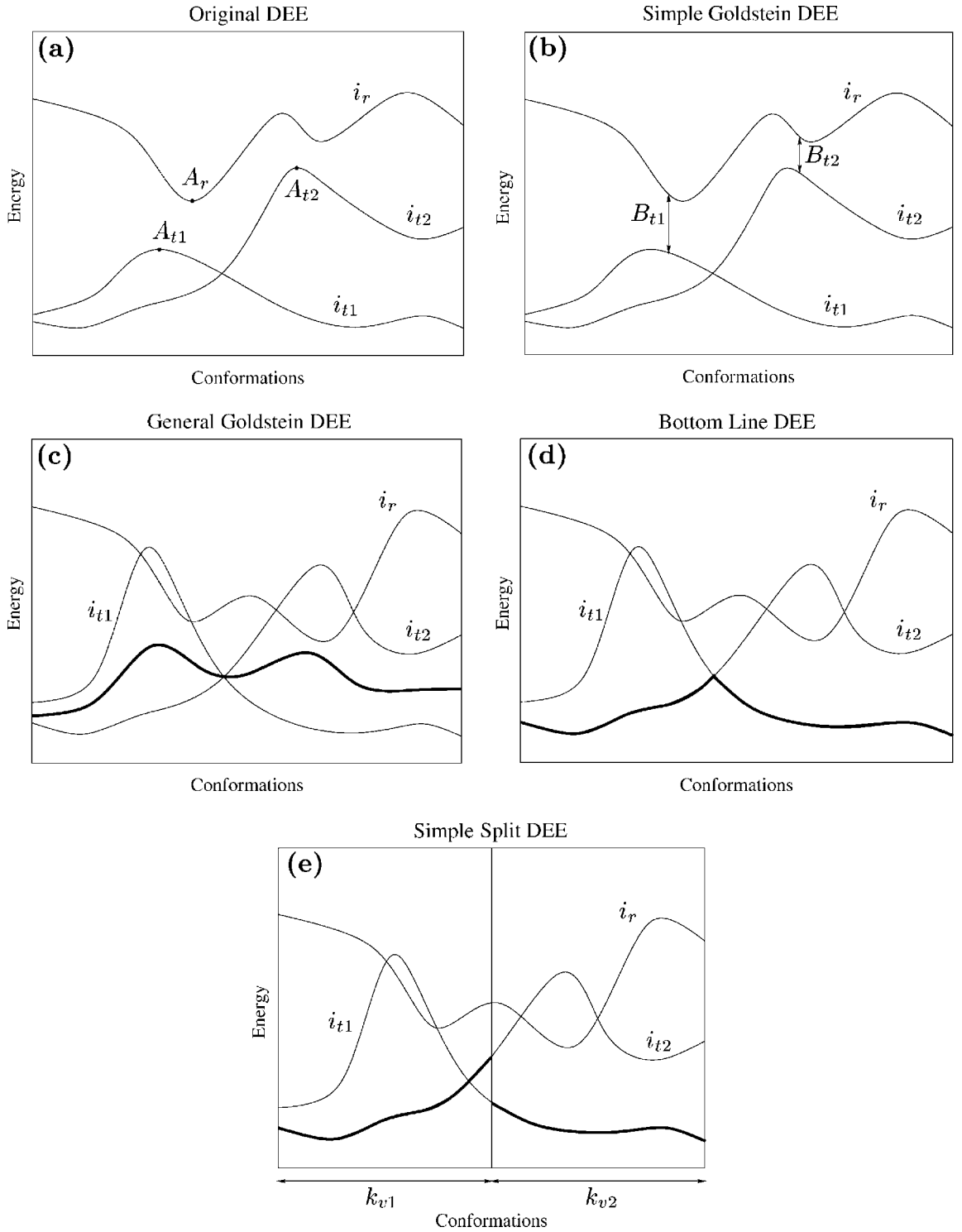

FIGURE 2. Different dead-end elimination criteria for sample energy profiles. The abscissa represents all possible conformations of the protein and the ordinate describes the net energy contribution produced by interactions with specific rotamers at position *i*. (a) Original DEE: i_r is eliminated by i_{t1} but not by i_{t2}. (b) Simple Goldstein DEE: i_r is eliminated by either i_{t1} or i_{t2}. (c) General Goldstein DEE: i_r cannot be eliminated by either i_{t1} or i_{t2}, but can be eliminated by a weighted average of the two. (d) Bottom line DEE: theoretically, i_r can be eliminated if the minimum of the i_{t1} and i_{t2} profiles always falls below the i_r profile. (e) Simple split DEE: i_r is eliminated by i_{t1} and i_{t2} in the partitions corresponding to splitting rotamers k_{v2} and k_{v1}, respectively.

GENERAL GOLDSTEIN DEE ($T = 1, \ldots, n-1$)

To attempt to cope with this more challenging scenario, Goldstein proposed a more general criterion:

$$E(i_r) - \sum_{t=1,T} C_t E(i_t) + \sum_{j,\, j \neq i} \left\{ \min_u \left[E(i_r j_u) - \sum_{t=1,T} C_t E(i_t j_u) \right] \right\} > 0 \quad (4)$$

which uses a weighted average of T candidate i_t rotamers to attempt to eliminate i_r (for notational convenience, T replaces the R originally used by Goldstein).[7] If some average of candidates is always better than the energy produced by i_r, it then follows that there is always at least one candidate i_t that provides a better alternative to i_r, regardless of which conformation turns out to be the GMEC. Figure 2c illustrates a case for which i_r can be eliminated by the average of i_{t1} and i_{t2} with constants $C_{t1} = C_{t2} = 1/2$. One complication with this approach is that the choice of constants is not obvious, although Lasters et al.[8] demonstrated that linear programming can be used to efficiently select suitable weights C_t. However, even if the optimal weights can be identified, this criterion is still more conservative than the most general idea underlying DEE. In a case in which none of the candidate i_t rotamers have lower energies than i_r for all conformations (and hence cannot eliminate i_r by themselves), then they will necessarily be raising the average of the hybrid somewhere in conformational space.

BOTTOM LINE DEE

Theoretically, it is unnecessary for the same i_t rotamer to eliminate i_r for all regions of conformational space. Instead, rotamer i_r can be eliminated if at least one candidate rotamer (but not necessarily always the same one) produces a lower energy for each possible conformation. As illustrated in Figure 2d, rotamer i_r could thus be eliminated if the "bottom line"[11] taken as the minimum energy of all the other possible alternative i_t candidates was always less than that produced using i_r. Although the bottom line criterion is theoretically the most powerful DEE elimination criterion, it is not apparent how to implement the approach efficiently. In particular, while the pictorial representations are helpful for visualizing the energy landscapes of the various candidate rotamers, these landscapes are never explicitly computed during the DEE procedure so it is impossible to overlay i_t profiles for joint comparison with the current i_r profile.

SIMPLE SPLIT DEE ($s = 1$)

As a next best alternative, the present work defines a means of splitting the conformational space into partitions, within which each of the candidate i_t rotamers can be compared singly with i_r. If at least one i_t rotamer produces a lower energy for each partition, then i_r can be eliminated even if no single i_t satisfies the simple Goldstein criterion [eq. (3)] for all partitions. Hence, for suitably defined partitions, it would be possible for rotamers i_{t1} and i_{t2} of Figure 2d to jointly eliminate rotamer i_r, even though neither of them could accomplish the feat alone. The difficulty is then finding a means of defining the partitions, because, in general, the ordering of the conformations along the abscissa of Figure 2d is arbitrary.

The partitioning approach adopted here is to split the conformational space into $O(n)$ equally sized partitions using the rotamers at some position $k \neq i$ (see Fig. 1b for notation). The split DEE criterion then becomes:

$$E(i_r) - E(i_t) + \sum_{j,\, j \neq k \neq i} \left\{ \min_u \left[E(i_r j_u) - E(i_t j_u) \right] \right\} + \left[E(i_r k_v) - E(i_t k_v) \right] > 0 \quad (5)$$

stating that i_r can be eliminated if, for each splitting rotamer v, at some splitting position $k \neq i$, there exists an i_t rotamer that yields a lower net energy contribution for all conformations within that partition. The splitting position k is thus removed from the summation in simple Goldstein DEE [eq. (3)] so that the relative merits of i_r and the various i_t candidates can be evaluated for each of the splitting rotamers k_v corresponding to individual partitions of the conformational space. As an example, Figure 2e illustrates a case in which i_r is successfully eliminated by splitting the conformation space into two partitions corresponding to the splitting rotamers k_{v1} and k_{v2} of Figure 1b. In effect, the splitting procedure expands the combinatorial space at one position to better leverage the candidate i_t rotamers in seeking to eliminate i_r.

GENERAL SPLIT DEE ($s = 1, \ldots, p-1$)

A larger fraction of the combinatorial space can be expanded by using s splitting locations simultaneously, corresponding to $O(n^s)$ partitions. The general split DEE criterion then becomes: elimi-

nate i_r if for all unique combinations of v at some $k_1 \ldots k_s \neq i \; \exists \; i_t$ s.t.:

$$E(i_r) - E(i_t) + \sum_{j,\, j \neq k_1 \ldots k_s \neq i} \min_u \left[E(i_r j_u) - E(i_t j_u)\right] + \sum_{k = k_1 \ldots k_s} \left[E(i_r k_v) - E(i_t k_v)\right] > 0 \quad (6)$$

As the number of splitting positions and partitions increases, so too does the cost per iteration.

COST BOUNDS

While the increasing power of these approaches is amply illustrated by a sequence of relevant energy landscapes, the increasing costs are not so readily apparent. In assessing the merits of each of the criteria, it is imperative to measure the elimination properties relative to the computational complexity of the method.

The complexity of each criterion is most simply defined in terms of the cost of each of the nested loops required to implement the approach. In all cases, both the self and pairwise energies are precomputed and stored, so that the complexity estimates do not take into account this preprocessing overhead.

For the original DEE theorem [eq. (2)], implementation of the left-hand side requires four nested loops: two over all positions i and j (each of length p) and two over all rotamers r and u (each of length n). The situation is identical for the expression on the right-hand side, so the cost of one round of dead-end elimination at each residue position is $O(n^2p^2)$ pairwise evaluations.

Moving to the simple Goldstein criterion [eq. (3)], the number of nested loops increases to five, because r and t are now used in the same expression, and the total cost per round of DEE becomes $O(n^3p^2)$ pairwise evaluations. The general Goldstein criterion introduces up to an extra $O(n)$ due to the summation over C_t, but this extra complexity can be avoided at the cost of a small additional memory overhead by precomputing these weighted sums.

The bottom line criterion does not lend itself to efficient implementation, requiring the evaluation of contributions using i_r and each i_t for every possible conformation. Hence, its implementation would have a complexity larger than $O(n^p)$, which is the total number of conformations.

For simple split DEE [eq. (5)], the cost per round of elimination remains $O(n^3p^2)$ as for simple Goldstein DEE. The implementation required to obtain this complexity is described later. For general split DEE, the cost is $O(n^{2+s}pq)$ pairwise evaluations per round of elimination, where q is the number of ways to choose s splitting positions:

$$q = \frac{(p-1)!}{s!(p-1-s)!} \quad (7)$$

If $s = 1$, this bound reduces to the $O(n^3p^2)$ of simple splitting. Using two splitting positions simultaneously ($s = 2$), the cost increases to $O(n^4p^3)$, and with $s = 3$, the complexity becomes $O(n^5p^4)$. The limit $s = p - 1$ corresponds to solving the entire combinatorial problem at cost $O(n^{p+1}p)$ pairwise evaluations.

Enhancements

The singles criteria just described form the heart of the dead-end elimination approach. However, there are a number of refinements based on this foundation that substantially increase the performance of the algorithm.

DOUBLES CALCULATIONS

The most important of these is the use of doubles criteria to flag dead-ending pairs for more efficient elimination using the singles criteria.[2, 6] These have been described previously for original and Goldstein DEE[2, 6, 8, 10] and will not be repeated here. The main point is that rotamer pairs (i_r, j_u) that have been identified as dead-ending need not be included in the minimization over u during subsequent rounds of singles elimination. This modification of the basic singles approach greatly increases the number of rotamers that can be eliminated, but also increases the complexity of the algorithm relative to the corresponding singles criterion by a factor of $O(n^2p)$. Hence, for the simple Goldstein criterion, the cost of one round of doubles elimination is $O(n^5p^3)$ pairwise evaluations.

MAGIC BULLET DOUBLES CALCULATIONS

Gordon and Mayo[10] introduced further modifications to the doubles procedure to yield substantial reductions in computational expense. First, they propose a "magic bullet doubles" version of the Goldstein criterion that has a reduced complexity of $O(n^4p^3)$. For the same complexity as a full application of original doubles, this approach identifies roughly 20% to 60% more dead-ending pairs. Furthermore, while their implementation of full Goldstein doubles remained $O(n^5p^3)$, they described simplifications that effectively introduced a small constant into this bound.[10]

MAGIC BULLET SPLITTINGS

When considering the use of split singles DEE, it is therefore advantageous to keep the complexity well below $O(n^5p^3)$, so as to keep the cost of singles elimination a small fraction of the more expensive doubles process. The complexity estimate for split singles [eq. (7)] reveals that the cost grows rapidly as s is increased. For $s = 1$, the cost is $O(n^3p^2)$, which is identical to the estimate for simple Goldstein singles. To reduce the cost for $s > 1$, it is desirable to use only the one "magic bullet" splitting $k_1 \ldots k_s$ that appears most likely to eliminate rotamer i_r; in this case, $q = 1$, so the cost of magic bullet split DEE reduces to $O(n^{2+s}p)$.

Intuitively, the best splitting locations should be those that have strong interactions with the rotamers at i. The objective is to find splitting positions $k_1 \ldots k_s$, such that the candidate i_t rotamers have widely varying interaction energies with rotamers at these positions $k_1 \ldots k_s$. Assuming no single i_t rotamer is able to eliminate i_r, the splitting will then enable other candidate i_t rotamers to contribute to the elimination of i_r.

For each i_r, the following metric can be used to rank the positions $k_1 \ldots k_s$ with which the i_t rotamers have the strongest adverse interactions:

$$\min_k \min_t \min_v \left[E(i_r k_v) - E(i_t k_v)\right] \qquad (8)$$

For $s = 2$ magic bullet splitting ($s = 2_{\mathrm{mb}}$), rotamers at the top two ranked positions are then used to split the conformational space, corresponding to a complexity of $O(n^4p)$. Using magic bullet splitting pairs is thus only a factor of $O(n/p)$ more expensive than using $s = 1$ splitting performed at all positions. Magic bullet splitting triplets ($s = 3_{\mathrm{mb}}$) with a cost bound of $O(n^5p)$ are also less expensive than Goldstein doubles, but in practice, the payoff of this approach does not appear to justify the mounting expense.

SPLITTING IMPLEMENTATION

For comparison purposes, the loop structures for efficient implementation of simple Goldstein singles DEE ($T = 1$) and simple split DEE ($s = 1$) are illustrated in Figure 3. Note that all of the self and pairwise energy terms are precomputed prior to invoking dead-end elimination. Using the present implementation, the complexity of both methods is $O(n^3p^2)$. Here, we made use of the fact that, for protein design calculations, the number of rotamers per position is much larger than the number of positions ($n > p$). Using split singles, there are two sets of nested loops at levels three to five so a single iteration of the algorithm is more costly than a single iteration using the Goldstein criterion. The key point for the split implementation is that the minima over u must be computed prior to the splitting over k to avoid additional nesting and a higher complexity.

Using an alternative formulation that accomplishes this same reduction in complexity, the summation over j, $j \neq i$ could be computed prior to splitting. Then the term corresponding to $\min_u$ at k could be subtracted out before adding back the

(a) Simple Goldstein singles

```
for each position i
   for each rotamer r at i
      for each candidate rotamer t at i
         X = E(i_r) − E(i_t)
         for all other positions j, j ≠ i
            for all rotamers u at position j
               Y = min_u[E(i_r j_u) − E(i_t j_u)]
            end
            X = X + Y
         end
         if (X > 0) eliminate i_r and return
      end
   end
end
```

(b) Simple split singles

```
for each position i
   for each rotamer r at i
      for each candidate rotamer t at i
         for all other positions j, j ≠ i
            for all rotamers u at position j
               store Y_jt = min_u[E(i_r j_u) − E(i_t j_u)]
            end
         end
      end
      for each splitting position k
         set elim_v = false ∀ v at k
         for each competitor t at i
            X = E(i_r) − E(i_t)
            for all other positions j, j ≠ i ≠ k
               X = X + Y_jt
            end
            for each partition v at k
               if ((X + [E(i_r k_v) − E(i_t k_v)]) > 0) elim_v = true
            end
         end
         if (elim_v = true ∀ v at k) eliminate i_r and return
      end
   end
end
```

FIGURE 3. Loop structure for implementing (a) Goldstein ($T = 1$) DEE and (b) split ($s = 1$) DEE. Both algorithms have the same $O(n^3p^2)$ complexity, as evident from the nesting of the loops of length n rotamers and p positions.

$[E(i_r k_v) - E(i_t k_v)]$ term. However, the subtraction step causes this approach to succumb to rounding errors even using double precision, while the formulation described in Figure 3 functions properly using single precision floating-point arithmetic.

The implementation just described is tailored toward protein design calculations where $n > p$. For side-chain placement calculations arising in homology modeling, it is more likely that $n < p$, because the rotamers represent only one amino acid identity at each position. In this case, the loop structure may be optimized differently to reflect the change in this relationship.[11]

DEE CYCLE

Regardless of the DEE criterion used, it is generally beneficial to apply the condition iteratively, as previous eliminations often facilitate the elimination of further rotamers. Eventually, no further rotamers will be eliminated at any position by additional rounds of dead-end elimination. At this point, it is necessary to resort to a doubles calculation to flag dead-ending pairs, which can then be used to increase the effectiveness of the singles elimination criterion. After further applications of a singles elimination criterion, it again becomes impossible to eliminate further rotamers. At this point, the rotamers at two positions are "unified" to form a superresidue that is treated as a single position for the remainder of the calculation.[7] This process permanently expands a fraction of the combinatorial space and sets off a new cascade of singles eliminations. Furthermore, the two unified positions are chosen to be those with the highest fraction of dead-ending pairs.[7, 10] These pairs become dead-ending superrotamers of the new superresidue and can thus be eliminated at the time of unification.

An efficient cycle of dead-end elimination procedures for protein design is described in Figure 4. Exhaustive Goldstein singles is followed by exhaustive $s = 1$ split singles followed by one round of $s = 2_{\mathrm{mb}}$ singles followed by one round of magic bullet Goldstein doubles. This process is then repeated taking advantage of the newly found dead-ending pairs. On the second time through the cycle, a full Goldstein doubles calculation is then performed instead of the magic bullet doubles approach. At the end of the third cycle, unification is performed in lieu of a doubles calculation and the whole process begins again. For purposes of this study, the complete cycle, as just described, will be referred to as split ($s = 2_{\mathrm{mb}}$) DEE. If the $s = 2_{\mathrm{mb}}$ splitting step is skipped, then the algorithm will be termed split ($s = 1$) DEE, and if neither splitting step is incorporated the method will be referred to as Goldstein ($T = 1$) DEE.

1) Simple Goldstein singles DEE ($T = 1$) until no further eliminations
2) Simple split singles DEE ($s = 1$) until no further eliminations
3) Magic bullet split singles DEE ($s = 2_{\mathrm{mb}}$) once for each rotamer
4) Alternate sequentially between the following, applying one during each cycle:
   - Fast Goldstein doubles calculation ($T = 1$) to flag dead ending pairs
   - Goldstein doubles calculation ($T = 1$) to flag dead ending pairs
   - Unification of two residues with highest fraction of dead ending pairs
5) return to 1

FIGURE 4. The preferred progression of dead-end elimination criteria for protein design.

Method

POTENTIAL FUNCTION

For protein design, the potential function incorporates terms for van der Waals interactions, hydrogen bonds, Coulomb interactions, and solvation, all of which have been described extensively in previous work.[9, 12–14] The solvation model is applied to core and boundary residues but not to surface residues.

ROTAMER LIBRARY

The backbone-dependent rotamer libraries used for protein design are also well documented,[9, 12] and are based on expansions of the χ_1 and χ_2 angles around the mean values from the Dunbrack and Karplus library.[5]

PARALLELIZATION

For protein design calculations, the DEE algorithm is parallelized within the outer two loops (i and r) of Figure 3; all loops nested within these are processed in parallel for different combinations of i and r. No communication is required between these subprocesses, because eliminations of different i_r rotamers can be performed independently. As a result, the DEE algorithm leads to very high parallel efficiencies on problems of practical interest.[10] The benchmark calculations were performed on eight R10000 processors of an SGI Origin 2000 running at 195 MHz.

BENCHMARK DESIGN CASES

To demonstrate the increased power of split DEE over standard Goldstein DEE, the present study presents results for the three challenging design

TABLE I.
Protein Design Benchmark Cases.

Case	Description	Type	Residues	Rotamers	Conformations
1	Plastocyanin	Core	18	806	2.4×10^{26}
2	β1 of Protein G	Core/boundary	19	2549	2.3×10^{37}
3	β1 of Protein G	Surface	14	2406	1.5×10^{31}

problems described in Table I. Each of the cases involves a different protein region, and each of these design problems is currently under computational and experimental study in the lab of one of the authors (S.L.M.). All methods described herein are exact in the sense that, when they do converge successfully, the same GMEC conformation is identified, regardless of the convergence path and elimination criteria employed.

Case 1 represents the design of 18 residues (5, 14, 21, 27, 29, 31, 37, 38, 39, 41, 72, 74, 80, 82, 84, 92, 96, 98) in the core of plastocyanin (PDB code 2pcy). Case 2 involves the design of all ten nonglycine core residues (3, 5, 7, 20, 26, 30, 34, 39, 52, 54) and nine boundary residues (1, 12, 23, 33, 37, 43, 45, 50, 56) of the β1 domain of Protein G (PDB code 1pga). Case 3 represents the design of 14 surface residues (4, 6, 8, 13, 15, 17, 42, 44, 46, 48, 49, 51, 53, 55) on the β-sheet of Protein G. For these designs, core residue identities are selected from among the amino acids A, V, L, I, F, Y, and W, whereas surface residue identities are selected from among A, N, Q, S, T, H, D, E, K, and R. Boundary residues are allowed to have amino acid identities from the union of these sets.

Results

BACKGROUND

For side-chain placement calculations, dead-end elimination algorithms are capable of determining the GMEC conformation for several hundred residues using well-resolved rotamer libraries. The number of rotamers per position is substantially higher for protein design calculations, because the conformational space contains rotamers for multiple amino acid identities for each position in the design. As a result, the computational efficiency and robustness of DEE are put to a much more challenging test and the number of positions that can be designed simultaneously is closer to a few dozen. The exact number is context dependent, because, for example, it is easier to eliminate rotamers in the core than on the surface due to the disparity in the strength of the interactions.

Typically, successful DEE calculations are characterized by a period of rapid elimination followed by a plateau and then a second period of rapid elimination leading to the GMEC conformation. The plateau occurs as the singles elimination criteria become less effective and more time is consumed searching for dead-ending pairs using doubles calculations. Unification of rotamers at multiple positions expands a portion of the conformational space and helps lead to new eliminations, but at the cost of temporarily increasing the number of rotamers in the calculation. This in turn aggravates the higher order dependence of the DEE complexity bounds on n, the number of rotamers at each position. To prevent the calculation from overrunning the available physical memory on the machine, a hard limit is placed on the maximum allowable number of rotamers. If the singles elimination criteria are unsuccessful in eliminating enough rotamers after each round of unification, the combinatorial buildup of superrotamers will eventually encounter this cutoff and the calculation will be forced to terminate. More powerful DEE criteria help to delay the onset of this buildup, thus allowing the simultaneous design of larger numbers of residues.

When comparing the computational efficiency of split DEE to standard Goldstein DEE, it is not possible to determine a speed-up factor that is relatively constant across all calculations. For easy design problems with few positions, both methods will converge rapidly in about the same amount of time; the extra cost per cycle in the split approach is balanced by the increased elimination power of the method. As the difficulty of the design problem increases, the speed-up provided by the split approach also increases, until, for some number of design positions, the standard DEE approach fails to converge and the speedup effectively becomes infinite. Eventually, for sufficiently large design calculations, the split approach will also fail to converge.

TABLE II.
CPU Minutes Consumed Using Goldstein ($T = 1$) DEE, Split ($s = 1$) DEE, and Split ($s = 2_{mb}$) DEE for Each of Three Test Cases.

Case	Method	($T = 1$) time	($s = 1$) time	($s = 2_{mb}$) time	Doubles time	Total time
1	Goldstein ($T = 1$)	108.4	—	—	299.4	418.5[a]
	Split ($s = 1$)	1.2	2.0	—	7.8	11.6
	Split ($s = 2_{mb}$)	1.1	2.2	0.9	7.1	11.8
2	Goldstein ($T = 1$)	660.2	—	—	1114.1	1793.4[a]
	Split ($s = 1$)	7.4	26.0	—	198.0	234.3
	Split ($s = 2_{mb}$)	5.9	21.3	13.0	175.7	219.2
2	Goldstein ($T = 1$)	1981.3	—	—	978.2	3005.8[a]
	Split ($s = 1$)	1338.4	3037.3	—	1062.0	5479.5[a]
	Split ($s = 2_{mb}$)	229.1	522.8	713.7	458.2	1939.0

[a] Failed to converge due to combinatorial explosion in the number of superrotamers created by unification.

BENCHMARK COMPUTATIONS

Timing results for the three benchmark design cases are provided in Table II with corresponding convergence histories shown in Figures 5–7. For the core design of case 1 (see Fig. 5), split ($s = 1$) and split ($s = 2_{mb}$) DEE converge to the GMEC conformation in under 12 minutes. By contrast, Goldstein ($T = 1$) DEE reaches a plateau with 4.5×10^{11} conformations remaining, and is eventually forced to terminate after 418 minutes when combinatorial buildup via unification causes the maximum allowable number of rotamers ($np_{max} = 10^4$) to be surpassed.

For the core/boundary design of case 2 (see Fig. 6), the standard Goldstein ($T = 1$) DEE algorithm plateaus at 1.1×10^{13} conformations before terminating due to combinatorial buildup ($np_{max} = 10^4$) after 1793 minutes. By contrast, split ($s = 1$) DEE converges to the GMEC conformation in 234 minutes and split ($s = 2_{mb}$) DEE converges slightly faster in 219 minutes.

For the surface design of case 3, the rotamers interact weakly relative to interactions in the core and boundary. Using the same maximum allowable number of rotamers as before ($np_{max} = 10^4$), split ($s = 2_{mb}$) DEE converge successfully in 2167 minutes whereas both Goldstein ($T = 1$) and split ($s = 1$) DEE quickly overrun the maximum rotamer limit (not shown). To observe a longer convergence path for these two algorithms, the maximum rotamer limit was increased ($np_{max} = 2 \times 10^4$) and the results are shown in Figure 7. Using Goldstein

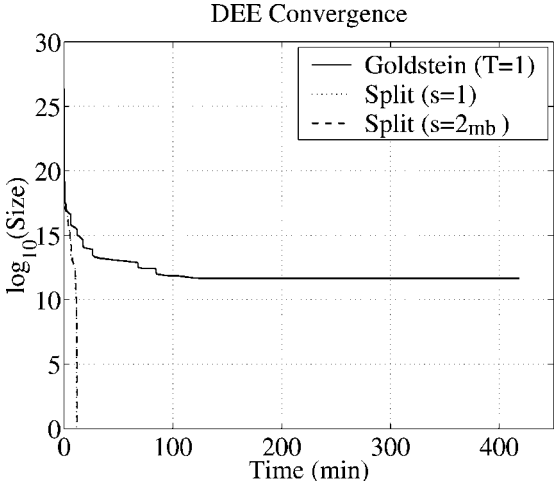

FIGURE 5. Plastocyanin core design (the two split methods are indistinguishable).

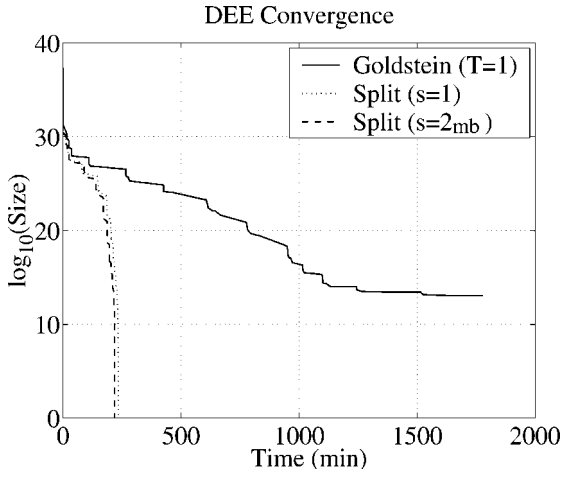

FIGURE 6. Protein G core/boundary design.

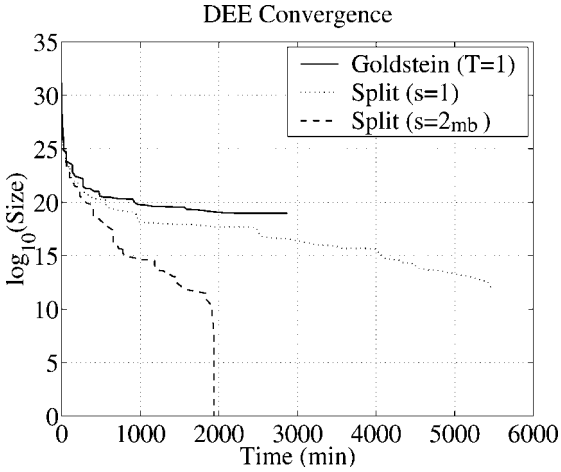

FIGURE 7. Protein G surface design.

($T = 1$) DEE, a plateau is reached at 9.3×10^{18} conformations and the calculation terminates due to rotamer buildup after 3006 minutes. Using split ($s = 1$) DEE, the number of conformations is reduced to 6.4×10^{11} before the calculation is terminated after 5480 minutes. Convergence to the GMEC conformation is achieved only with split ($s = 2_{mb}$) DEE, requiring 1939 minutes. For the hardest problems, which involve weak interactions between surface residues, the more powerful ($s = 2_{mb}$) criterion can lead to substantial improvements in the overall performance of the algorithm, even relative to split ($s = 1$) DEE.

Conclusions

Conformational splitting criteria significantly increase the power of dead-end elimination algorithms for the purposes of sequence selection in computational protein design. For challenging design calculations, the two splitting methods ($s = 1$) and ($s = 2_{mb}$) dramatically increase the efficiency of DEE relative to existing state-of-the-art methods based on Goldstein ($T = 1$) singles elimination. Although the two split DEE methods perform similarly for the design of core and boundary residues, the more powerful split ($s = 2_{mb}$) algorithm can provide significant advantages for calculations involving weakly interacting surface residues. Using this improved method, it is now possible to perform protein design calculations that were previously intractable using reasonable time and memory allocations on current supercomputers.

Acknowledgments

N.A.P. thanks D. B. Gordon for many helpful discussions. J.A.S. and J.D. thank I. Lasters and M. De Maeyer for their interest and occasional help as peers in the common field.

References

1. Dahiyat, B. I.; Mayo, S. L. Science 1997, 278, 82.
2. Desmet, J.; De Maeyer, M.; Hazes, B.; Lasters, I. Nature 1992, 356, 539.
3. Janin, J.; Wodak, S.; Levitt, M.; Maigret, D. J Mol Biol 1978, 125, 357.
4. Ponder, J. W.; Richards, F. J Mol Biol 1987, 193, 775.
5. Dunbrack, R. L.; Karplus, M. J Mol Biol 1993, 230, 543.
6. Lasters, I.; Desmet, J. Prot Eng 1993, 6, 717.
7. Goldstein, R. F. Biophys J 1994, 66, 1335.
8. Lasters, I.; De Maeyer, M.; Desmet, J. Prot Eng 1995, 8, 815.
9. Dahiyat, B. I.; Mayo, S. L. Prot Sci 1996, 5, 895.
10. Gordon, D. B.; Mayo, S. L. J Comput Chem 1998, 19, 1505.
11. Desmet, J.; De Maeyer, M.; Lasters, I. In: Altman, R. B., et al., eds. Pacific Symposium on Biocomputing '97; World Scientific: Singapore, 1997; p 122.
12. Dahiyat, B. I.; Gordon, D. B.; Mayo, S. L. Prot Sci 1997, 6, 1333.
13. Street, A. G.; Mayo, S. L. Folding Des 1998, 3, 253.
14. Gordon, D. B.; Marshall, S. A.; Mayo, S. L. Curr Opin Struct Biol 1999, 9, 509.