A Method for Encoding the HP Model in D Dimensions

William Macready
D-Wave Systems Inc.

January 17, 2008

1 Preliminaries

Let $S = (s_1, \cdots, s_N)$ represent the primary structure with $s_i \in \{\mathrm{H}, \mathrm{P}\}$. Let $\mathbf{x}_i = (x_i^1, x_i^2, \cdots, x_i^D)$ represent the lattice location in D dimensions of the ith amino acid in the primary structure. Collectively, we refer to all $\mathbf{x}_i$ as $\mathbf{X}$. If we consider a D-dimensional hypercubic lattice (other lattices will be similar), then the domain of each x_i^d is $\{0, \cdots, N-1\}$. We represent each value in binary using $\lceil \log_2 N \rceil$ Boolean variables. If N is not a power of 2, then we do not need to add a constraint that says the binary representation of x_i^d is less than N because other constraints will enforce legal folds. We define $n = \lceil \log_2 N \rceil$.

We write x_i^d in terms of Boolean variables as

$$x_i^d = \sum_{\alpha=0}^{n-1} 2^\alpha b_{i,\alpha}^d$$

where $b_{i,\alpha}^d$ is the α bit in the binary representation of x_i^d.

It will be convenient for subsequent modeling to define a 1-to-1 function $f(\mathbf{x})$ that maps a lattice location to an integer. For hypercubic lattices we can take[1]

$$f(x_i^1, x_i^2, \cdots, x_i^D) = x_i^1 + N x_i^2 + \cdots + N^{D-1} x_i^D.$$

We also view f as a function of the nD Boolean variables $b_{i,\cdot}^{\cdot}$. Further, on hypercubic lattices we observe that two lattice locations $\mathbf{x}_i$ and $\mathbf{x}_j$ are neighbours iff

$$|f(\mathbf{x}_i) - f(\mathbf{x}_j)| = 1 \quad \text{or} \quad |f(\mathbf{x}_i) - f(\mathbf{x}_j)| = N \quad \text{or} \quad \cdots \quad \text{or} \quad |f(\mathbf{x}_i) - f(\mathbf{x}_j)| = N^{D-1}. \tag{1}$$

2 The energy function

There are three terms in the energy function to account for:

1. self-avoidance: making sure that the tertiary structure does not represent a fold that doubly occupies a lattice site
2. valid walk: making sure that the tertiary structure is connected
3. pairwise potential: counting the number of neighbouring H-H interactions in the tertiary structure

[1] It is necessary that this function is linear in its arguments.

Terms 1 and 2 are hard constraints and 3 enters as the optimization objective.

Before we derive energy contributions accounting for each requirement we note the symmetries of the problem. Any fold can be rigidly translated and acted upon by any element from the symmetry group of the lattice, and this will preserve the energy of the fold. We would like to eliminate all such symmetries, and we can eliminate translations and rotations by clamping a pair of neighbouring amino acids to a specific pairs of adjacent lattice locations. This means that any primary structure of length N is effectively only of length $N-2$. We clamp the middle of the primary sequence, amino acids $s_{i^*} \equiv s_{\lfloor N/2 \rfloor}$ and $s_{i^*+1} \equiv s_{\lfloor N/2 \rfloor + 1}$ to lattice locations $\mathbf{x}_{i^*} = (\lfloor N/2 \rfloor, \lfloor N/2 \rfloor, \cdots, \lfloor N/2 \rfloor)$ and $\mathbf{x}_{i^*+1} = (\lfloor N/2 \rfloor + 1, \lfloor N/2 \rfloor, \cdots, \lfloor N/2 \rfloor)$

2.1 Self-avoidance

The self avoidance requirement can be enforced by requiring that no pair of amino acids have the same $f(\mathbf{x})$ value. More formally, we require $\forall i,j \; f(\mathbf{x}_i) \neq f(\mathbf{x}_j)$ or equivalently[2]

$$\forall i,j \; \big(f(\mathbf{x}_i) \leq f(\mathbf{x}_j) - 1 \vee f(\mathbf{x}_j) \leq f(\mathbf{x}_i) - 1\big).$$

This generates $\binom{N}{2} - 1$ constraints each of which is a disjunction of linear inequalities.[3] We represent the disjunction of linear inequalities as follows: For each i,j we introduce a Boolean variable $c_{i,j}$, and write the disjunction of constraints as the conjunction

$$\big(f(\mathbf{x}_i) - f(\mathbf{x}_j) + 1 \leq N^D c_{i,j}\big) \wedge \big(f(\mathbf{x}_j) - f(\mathbf{x}_i) + 1 \leq N^D (1 - c_{i,j})\big)$$

where N^D is an upper bound on the value of both $f(\mathbf{x}_i) - f(\mathbf{x}_j) + 1$ and $f(\mathbf{x}_j) - f(\mathbf{x}_i) + 1$. If $c_{i,j} = 1$ the first constraint is trivially satisfied as N^D is an upper bound, and the second constraint is active. If on the other hand $c_{i,j} = 0$, then the first constraint is active, and the second is trivially satisfied.

Next, we convert the inequality constraints to equality constraints. We write the constraints as

$$\big(f(\mathbf{x}_i) - f(\mathbf{x}_j) - N^D c_{i,j} \leq -1\big) \wedge \big(f(\mathbf{x}_j) - f(\mathbf{x}_i) + N^D c_{i,j} \leq N^D - 1)\big).$$

The maximal slackness of the left/right constraint is $2(N^D - 1)/2N^D$ respectively. Thus, for each i,j pair we need to introduce at most $\lceil \log_2 2N^D \rceil = 1 + Dn$ slack variables $s_{i,j;\alpha}$ representing the slackness. Further, since only one of the constraints can ever be active, then a single set of slack variables is sufficient. Thus, we can write the pair of inequality constraints as the pair of equality constraints

$$f(\mathbf{x}_i) - f(\mathbf{x}_j) - N^D c_{i,j} + \sum_{\alpha=0}^{Dn} 2^\alpha s_{i,j;\alpha} + 1 = 0$$

$$f(\mathbf{x}_j) - f(\mathbf{x}_i) + N^D c_{i,j} + \sum_{\alpha=0}^{Dn} 2^\alpha s_{i,j;\alpha} + 1 - N^D = 0$$

Note that the sums on α extend to include Dn, and do not stop at $Dn - 1$ since the slackness extends to $2N^D$. These equality constraints contribute as penalties to the objective by squaring the left hand side of each equality constraint. This will contribute $\binom{N}{2} - 1$ quadratic terms (in the sum over $i > j$ exclude the

[2]Note that both constraints cannot simultaneously be true.

[3]Recall that two amino acids are fixed to neighbouring lattice locations and do not need to be accounted for.

fixed pair):

$$E_1(X) = \lambda_1 \sum_{i>j} \Bigg\{ \Big(f(\mathbf{x}_i) - f(\mathbf{x}_j) - N^D c_{i,j} + \sum_{\alpha=0}^{n} 2^\alpha s_{i,j;\alpha} + 1 \Big)^2 + \Big(f(\mathbf{x}_j) - f(\mathbf{x}_i) + N^D c_{i,j} + \sum_{\alpha=0}^{n} 2^\alpha s_{i,j;\alpha} + 1 - N^D \Big)^2 \Bigg\}.$$

λ_1 is the multiplier adjusting the weight of this self-avoidance penalty.

Variable count: The total new variable count is $\big(\binom{N}{2}-1\big)(1+1+Dn) = \big(\binom{N}{2}-1\big)(Dn+2)$ to account for both the indicator variables $c_{i,j}$ (one per i,j pair), and the slack variables $s_{i,j;\alpha}$ ($1+Dn$ per i,j pair). There are $\binom{N}{2}-1$ pairs to check since the two mid-point clamped amino acids s_{i^*} and s_{i^*+1} do not need to be checked.

2.2 Valid walks

Valid walks require that neighbouring amino acids in the primary structure are neighbours on the lattice once mapped to their tertiary structure. Two lattice locations $\mathbf{x}_i$ and $\mathbf{x}_j$ are neighbours iff $\|\mathbf{x}_i - \mathbf{x}_j\|^2 = 1$. The Euclidean distance is clearly a quadratic function and can be represented without additional variables.

If the self avoidance constraint has been imposed with a large penalty then $\|\mathbf{x}_i - \mathbf{x}_j\|^2$ must be greater than or equal to 1. Thus, we may impose the penalty

$$E_2(X) = \lambda_2 \sum_{i=1|i\neq i^*}^{N-1} \|\mathbf{x}_i - \mathbf{x}_{i+1}\|^2$$

where λ_2 is sufficiently smaller that λ_1 to ensure that constraint 1 is always satisfied in a low energy state.

2.3 Counting H-H bonds

To count the number of H-H interactions[4] we use an indicator variable $r_{i,j}$ which if 1 implies that the ith and jth amino acids are neighbours on the lattice. With this definition we can count the number of H-H interactions by minimizing

$$-\sum_{i,j}{}' r_{i,j}$$

where the primed sum is restricted to those amino acids where $s_i = \mathrm{H}$ and $s_j = \mathrm{H}$, and where the separation $|i-j|$ is greater than 1 (so that they are not neighbours along the primary structure). On bipartite lattices, like the hypercubic lattice, we can further restrict things to amino acids both being H that are separated by $|i-j| = 3, 5, 7, \cdots$. It is only these possibilities that will allow for an H-H interaction.

Maximizing this term will set $r_{i,j}$ to 1, and in that case we want to trigger constraints that check if $\mathbf{x}_i$ and $\mathbf{x}_j$ are neighbours. If they are not, then the penalty is large enough so that $r_{i,j}$ will remain zero. If $r_{i,j} = 1$ then lattice $\mathbf{x}_i$ and $\mathbf{x}_j$ can be neighbours in $2D$ possible ways, one if which will be true. We indicate which

[4]The results of this section are straightforwardly extended to larger amino acid alphabets, and more complex pairwise interaction energies.

neighbour through the Booleans $d_{i,j;k}$ where k ranges over the $2D$ possible neighbours. Thus, we want to express the requirement that

$$r_{i,j} = 1 \to \sum_{k=1}^{2D} d_{i,j;k} = 1. \tag{2}$$

The $d_{i,j;k}$ variables turn on penalties checking for neighbours; that is we add terms

$$\begin{aligned} d_{i,j;1}\big(f(\mathbf{x}_i) - f(\mathbf{x}_j) - 1)\big)^2 + d_{i,j;2}\big(f(\mathbf{x}_i) - f(\mathbf{x}_j) + 1)\big)^2 + \\ d_{i,j;3}\big(f(\mathbf{x}_i) - f(\mathbf{x}_j) - N)\big)^2 + d_{i,j;4}\big(f(\mathbf{x}_i) - f(\mathbf{x}_j) + N)\big)^2 + \cdots \end{aligned} \tag{3}$$

The implication constraint Eq. (2) is modeled as

$$\sum_k d_{i,j;k} - r_{i,j} = 0.$$

When $r_{i,j} = 1$ this imposes the correct condition, and when $r_{i,j} = 0$ this requires all $d_{i,j;k}$ to be zero so that we don't check the neighbourhood penalty terms.

The implication constraint contributes to the object by squaring the left hand side of the above equation. We must also add the cubic penalty function Eq. (3). The total contribution to the optimization objective is

$$\begin{aligned} E_3(X) = \sum_{i>j}{}' \Bigg\{ & -r_{i,j} + \lambda_3^1\Bigg(\sum_{k=1}^{2D} d_{i,j;k} - r_{i,j}\Bigg)^2 + \\ & \lambda_3^2 d_{i,j;1}\big(f(\mathbf{x}_i) - f(\mathbf{x}_j) - 1)\big)^2 + \lambda_3^2 d_{i,j;2}\big(f(\mathbf{x}_i) - f(\mathbf{x}_j) + 1)\big)^2 + \\ & \lambda_3^2 d_{i,j;3}\big(f(\mathbf{x}_i) - f(\mathbf{x}_j) - N)\big)^2 + \lambda_3^2 d_{i,j;4}\big(f(\mathbf{x}_i) - f(\mathbf{x}_j) + N)\big)^2 + \cdots \Bigg\} \end{aligned}$$

The weights λ_3^1 and λ_3^2 adjust the importance of the penalty terms.

Variable count: We must linearize the quadratic terms in the squares appearing in Eq. (3) using variables $y^d_{i,j,\alpha;k} = d_{i,j;k} b^d_{i,\alpha}$.[5] For each i, j pair this adds $2nD \times 2D = 4nD^2$ variables. In addition there is the single $r_{i,j}$ and the $2D$ $d_{i,j;k}$ variables. Thus, if there are $\mathcal{H}(S)$ amino acids that enter the sum for some primary structure S, then the total number of new variables is $\mathcal{H}(S)(1 + 2D + 4nD^2)$.[6] This estimate is pessimistic because, depending on the primary structure, some possible i and j may recur in the set of possible pairs. In such cases we can reuse some of the previously defined pairs, but I will not account for such improvements here.

3 Total variable count

The total objective function is taken as

$$E(X) = E_1(X) + E_2(X) + E_3(X).$$

[5]Since we require that i, j are not neighbours in S, none of these new pairwise variables have previously been defined.

[6]$\mathcal{H}(S)$ counts the number of pairs in S where $s_i = \mathrm{H}$, $s_j = \mathrm{H}$, and $|i - j| = 3, 5, 7, \cdots$.

In addition to the $(N-2)nD$ base variables $b^d_{i,k}$ we have had to introduce other variables, the total count is

$$\underbrace{(N-2)nD}_{\text{base}} + \underbrace{\left(\binom{N}{2}-1\right)(Dn+2)}_{\text{self avoidance}} + \underbrace{\mathcal{H}(S)\Big(1+2D+4nD^2\Big)}_{\text{H-H counting}}.$$

In the worst case where $S=(\mathrm{H\ H\ H}\cdots\mathrm{H})$, i.e. all H, $\mathcal{H}(S)=\mathcal{O}(N^2)$. The order of each contribution is

$$\mathcal{O}(NnD)+\mathcal{O}(N^2nD)+\mathcal{O}(N^2nD^2).$$

The highest order term is $\mathcal{O}(N^2nD^2)$. For $N=32$, $D=3$ we have $n=5$ and $\mathcal{H}(S)=\binom{N-1}{2}=465$ on general lattices, and $\mathcal{H}(S)=240$ on bipartite lattices.[7] The variable count on a bipartite lattice is therefore

$$30(5)(3)+(496-1)(17)+240(1+6+180)=53745$$

This is the worst case. For example, for the primary structure PHHHPPHHHHPHHHHHHPHPHPPHHH-PPPHHH, which has 11 Ps, has $\mathcal{H}(S)=100$ (on bipartite lattices), and the total variable count is 27565.

4 Ideas for further reductions

Here are some ideas for further reducing the variable count and/or increasing the relative utility of this approach vs. classical approaches:

- Exploit geometric symmetries (further pin the fold so that possible degenerate folds are eliminated): might give prefactor difference in number of variables
- The H-H counting term as introduced here has cubic terms; a lot of variables are introduced to reduce to quadratic. It is likely that by doing something like was done in the self-avoidance encoding these cubic terms could be eliminated
- Apply these basic concepts to more realistic lattice models without increasing the number of variables (ie make the problem harder/more realistic but keep the variable count fixed by construction)

I suspect that the limit for encoding into binary variables here is limited by either $\mathcal{O}(N^2)$ or $\mathcal{O}(N\log N)$. If it's the former then we're up against the limit and gains are to be had by reducing the prefactor. To see if things can be done in $\mathcal{O}(N\log N)$ I would try to see if the self-avoidance term can be done in $\mathcal{O}(N\log N)$ variables somehow. If it can then I suspect the $H-H$ term can be also.

[7] I evaluated $\mathcal{H}(S)$ with Matlab code, I could figure out a closed form for the string of all H.