Proceedings of the 2007 IEEE Symposium on Artificial Life (CI-ALife 2007)

HIFF-II: A Hierarchically Decomposable Problem with Inter-level Interdependency

Susan Khor
Concordia University, Montreal, CANADA
slc_khor@cse.concordia.ca

***Abstract* – A new test problem called HIFF-II is presented. We propose that HIFF-II discriminates evolutionary algorithms that use recombination from those that do not. HIFF-II is a variant of the Hierarchical-If-And-Only-If (HIFF) problem proposed by Watson [1]. HIFF-II differs from HIFF in one significant way. The dependency matrix for HIFF-II is sparser than that for HIFF. Two important consequences from this are HIFF-II has inter-level interdependencies and there are distinct sets of optimal solutions of the same cardinality at every level.**

We tested the performance of a random mutation hill climbing algorithm and a genetic algorithm on HIFF-II. This experiment was conducted under "ideal" circumstances for each algorithm on the HIFF-II problem. For the random mutation hill climbing algorithm, this involved using an altruistic selection scheme to induce progress at all levels simultaneously and using a reasonably low mutation rate. For the genetic algorithm, this meant using the gene-invariant genetic algorithm (GIGA) which preserves population diversity throughout a run and selecting pairs of parents that are close in aggregate fitness value.

This experiment confirmed that random mutation hill climbers experienced more difficulty evolving an optimal solution for HIFF-II than a genetic algorithm. The disparity between the performances of the two algorithms became more apparent as the problem size increases.

I. INTRODUCTION

In a previous paper [2], we proposed a method to enhance the adaptive capability of a random mutation hill climbing (RMHC) algorithm [3] on the continuous Hierarchical-If-And-Only-If (HIFF) problem [4]. Our method involved the use of a *phenotype* and a *non-altruistic selection scheme*. The elements of a phenotype are made up of per-level fitness values of the genotype. The selection scheme does a pair-wise comparison of the elements of two phenotypes in some order, and selects the first genotype which confers an advantage. This selection scheme is non-altruistic because progress at one level can come at the cost of regress at another level in one phenotype. Section IV elaborates on the selection scheme.

While the results in [2] enhanced the argument that the blind process of evolution can produce complex forms, albeit abstractly, it once again blurred the distinction between the adaptive capabilities of evolutionary algorithms which use recombination, typified by genetic algorithms [5] and those which do not, typified by hill climbers. This debate about the utility of recombination in evolutionary algorithms is a long standing one [6].

In this paper, we present *HIFF-II*, a variant of the HIFF problem. Section II gives details on HIFF-II. HIFF-II differs from HIFF structurally – the dependency matrix capturing the interdependency between variables in HIFF-II is less dense than that for HIFF. This seemingly small structural difference has some interesting consequences for search difficulty as section III explains.

HIFF-II is difficult to solve by our previous method [2], but can be solved if an *altruistic selection scheme* is used in our previous method. Under an altruistic selection scheme, a variant is selected only when progress is achieved at all levels simultaneously. Setting aside the question whether an altruistic selection scheme is evolutionary, there remains the question of the scalability of this solution because constraints on all levels must be satisfied before any progress can be made.

As with HIFF, HIFF-II is solvable by a genetic algorithm which maintains population diversity. However, we found the Gene-Invariant Genetic Algorithm (GIGA) [7] to be more efficient on HIFF-II than a genetic algorithm with deterministic-crowding (GA-DC) [8]. Section V reports our experimental results on HIFF-II.

Our main findings are:

(i) If one considers an altruistic scheme to be un-evolutionary in the sense that natural selection favors short sighted gains, then an optimal solution to HIFF-II is difficult to evolve with a RMHC algorithm but is evolvable with a genetic algorithm. This conclusion holds even when RMHC is enhanced with phenotypes and a level directed (non-altruistic) selection scheme, and when GA-DC is used instead of GIGA.

(ii) If one is not concerned with the "evolutionariness" of a solution as is often the case in the engineering domain, then an optimal solution to HIFF-II is still difficult to evolve with a RMHC algorithm but is evolvable with a genetic algorithm. This conclusion holds when both the RMHC algorithm and the genetic algorithm are run under "ideal" conditions, and when the problem size (N) is significantly large. Through our experiments, we found $N \geq 256$ to be significantly large. "Ideal" conditions for a RMHC algorithm include using an altruistic selection scheme and a suitably low mutation rate. "Ideal" conditions for a genetic algorithm include gene-invariance (GIGA) and selecting pairs of genotypes with close fitness values to mate.

In summary, a RMHC algorithm has more difficulty evolving an optimal solution to HIFF-II than a genetic algorithm. On the basis of these findings, we offer HIFF-II as a test problem that highlights the utility of recombination in an evolutionary algorithm.

II. THE HIFF-II FUNCTION

This section defines the HIFF-II function and explains how per-level fitness values are calculated to form phenotypes. A

phenotype is a multi-dimensional representation of a genotype's (aggregate) fitness. The (aggregate) fitness of a genotype is split into a sequence of per-level fitness values and represented as a phenotype.

As in our previous work [2], fitness of a genotype is computed by levels. A genotype is a bit string of length $N = 2^n$. There are $\log_2 N$ levels and $N/2^{\lambda}$ modules of size 2^{λ} at level λ for a genotype and $\lambda = 1 \ldots n$. Table 1 illustrates for $N = 8$.

TABLE 1

Level (λ)	Number of Modules	Module size	Bit positions by module							
3 (highest)	1	8	0	1	2	3	4	5	6	7
2	2	4	0	1	2	3	0	1	2	3
1 (lowest)	4	2	0	1	0	1	0	1	0	1

Per-level fitness of a genotype is the sum of the fitness of every module at a level. Module fitness is obtained by comparing bits in the first half of a module with bits in the second half of a module. Fig.1 illustrates this process and Table 2 gives an example of a HIFF-II calculation. Specifically, fitness of a module at level λ is calculated as follows:

(i) gene i is compared with gene $2^{\lambda-1} + i$ of a module for $i = 0, \ldots, 2^{\lambda-1} - 1$,

(ii) one point is awarded if the two genes have the same allelic value, i.e. the bits are equal, and

(iii) the number of points is divided by half the module size, $2^{\lambda-1}$.

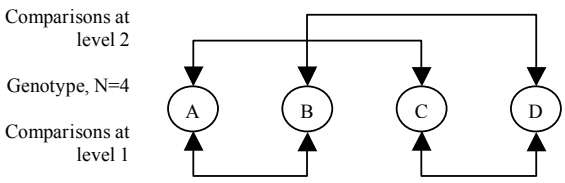

Fig. 1 Illustration of which genes in a genotype are compared to compute per-level fitness values

TABLE 2
AN EXAMPLE OF HIFF-II EVALUATION. LEFTMOST BIT OF A MODULE IS THE ZEROTH BIT.

Level	Bit Comparisons Per Module	Genotype 187								Per level fitness
		1	0	1	1	1	0	1	1	
3	(0, 4), (1, 5), (2, 6), (3, 7)					1	1	1	1	4/4 = 1
2	(0, 2), (1, 3)			1	0			1	0	1/2 + 1/2 = 1
1	(0, 1)		0		1		0		1	0/1 + 1/1 + 0/1 + 1/1 = 2
								Phenotype		$\langle 1, 1, 2 \rangle$
								Aggregate fitness		4

Like HIFF, HIFF-II has two global optima: the all zeroes and the all ones genotypes, the optimal aggregate fitness for a problem of size $N = 2^n$ is N-1 and the optimal phenotype is $\langle 2^0, 2^1, \ldots, 2^i, 2^{i+1}, \ldots 2^{n-1} \rangle$. Like HIFF, HIFF-II has a consistent hierarchical structure with interdependencies within and between modules as indicated in Fig. 1. However, unlike HIFF, HIFF-II has inter-level interdependencies so solutions optimal for one level need not be optimal for another higher or lower level. Section III explains.

III. COMPARISON OF HIFF-II WITH HIFF

A. Structure of interdependencies

One of the main features of the HIFF problem is its modular interdependency [4]. In a problem with modular interdependency, the variables of a problem are related in such a way that intra-module dependencies are stronger than inter-module dependencies but the inter-module dependencies are important enough that they must be resolved in order to find an optimal solution to the whole problem. In other words, a problem with modular interdependency is decomposable into modules but the modules are non-separable. If modules of a problem are separable, an optimal solution for each module can be found independently of the other modules in the problem and these partial optimal solutions can be aggregated linearly to obtain an optimal solution to the whole problem.

The two components of modular interdependency: structure of dependencies and strength of dependencies are reflected compactly in a dependency matrix. Fig. 2A shows the dependency matrix for HIFF-II with 8 variables. The dependency matrix for HIFF with 8 variables is produced in Fig. 2B for comparison (This matrix is a reproduction of the dependency matrix in [1, p.14] but each element of the matrix is divided by 32.).

	0	1	2	3	4	5	6	7
0	-	1/2	1/4	0	1/8	0	0	0
1	1/2	-	0	1/4	0	1/8	0	0
2	1/4	0	-	1/2	0	0	1/8	0
3	0	1/4	1/2	-	0	0	0	1/8
4	1/8	0	0	0	-	1/2	1/4	0
5	0	1/8	0	0	1/2	-	0	1/4
6	0	0	1/8	0	1/4	0	-	1/2
7	0	0	0	1/8	0	1/4	1/2	-

Fig. 2A Dependency matrix for HIFF-II with 8 variables

	0	1	2	3	4	5	6	7
0	-	1/2	1/8	1/8	1/32	1/32	1/32	1/32
1	1/2	-	1/8	1/8	1/32	1/32	1/32	1/32
2	1/8	1/8	-	1/2	1/32	1/32	1/32	1/32
3	1/8	1/8	1/2	-	1/32	1/32	1/32	1/32
4	1/32	1/32	1/32	1/32	-	1/2	1/8	1/8
5	1/32	1/32	1/32	1/32	1/2	-	1/8	1/8
6	1/32	1/32	1/32	1/32	1/8	1/8	-	1/2
7	1/32	1/32	1/32	1/32	1/8	1/8	1/2	-

Fig. 2B Dependency matrix for HIFF with 8 variables

The dependency matrices can be used to compute the HIFF-II and HIFF fitness values for a genotype. Fig. 3 illustrates how on genotype number 187 (1011 1011) which has a HIFF-II fitness value of 4 and a HIFF fitness value of 3.625. First, a work dependency matrix is prepared for the genotype in question. All non-diagonal entries in the work dependency matrix are set to zero. Then, where two distinct bits i and j of the genotype agree, the value of element (i, j) in the corresponding dependency matrix is copied into the work

dependency matrix. For example, bits 0 and 2 agree in genotype number 187, so the entry in the HIFF-II work dependency matrix at (0, 2) is 1/4. The sum of the entries in a completed work dependency matrix gives the fitness of the genotype.

	1	0	1	1	1	0	1	1
1	-	0	1/4	0	1/8	0	0	0
0	0	-	0	0	0	1/8	0	0
1	1/4	0	-	1/2	0	0	1/8	0
1	0	0	1/2	-	0	0	0	1/8
1	1/8	0	0	0	-	0	1/4	0
0	0	1/8	0	0	0	-	0	0
1	0	0	1/8	0	1/4	0	-	1/2
1	0	0	0	1/8	0	0	1/2	-

Fig. 3A Completed work dependency matrix to calculate HIFF-II fitness for genotype 1011 1011

	1	0	1	1	1	0	1	1
1	-	0	1/8	1/8	1/32	0	1/32	1/32
0	0	-	0	0	0	1/32	0	0
1	1/8	0	-	1/2	1/32	0	1/32	1/32
1	1/8	0	1/2	-	1/32	0	1/32	1/32
1	1/32	0	1/32	1/32	-	0	1/8	1/8
0	0	1/32	0	0	0	-	0	0
1	1/32	0	1/32	1/32	1/8	0	-	1/2
1	1/32	0	1/32	1/32	1/8	0	1/2	-

Fig. 3B Completed work dependency matrix to calculate HIFF fitness for genotype 1011 1011

However, what is relevant for our purpose here is the relative densities of the two dependency matrices in Fig. 2. The most striking difference is the prevalence of zeroes in the dependency matrix for HIFF-II and the absence of zeroes in the dependency matrix for HIFF. But this does not mean that there are independent variables in HIFF-II. Fig. 4 illustrates the structure and strength of the interdependencies present in a HIFF-II problem with 4 variables with a dependency graph.

	A	B	C	D
A	-	1/2	1/4	0
B	1/2	-	0	1/4
C	1/4	0	-	1/2
D	0	1/4	1/2	-

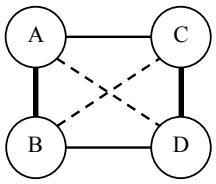

Fig. 4 Interdependencies between the 4 genes in HIFF-II. Solid lines show direct interdependencies and dashed lines show indirect interdependencies. Thickness of the solid lines indicates strength of the interdependency. Thicker solid lines indicate stronger interdependency. An equality between A and B is worth half a point but equality between A and C is only worth quarter of a point.

While variables B and C are not directly dependent on each other (there is no edge between B and C in Fig. 1), B and C are indirectly interdependent through their direct interdependence on variables A and D. Similarly, variables A and D are indirectly interdependent through their direct interdependencies on variables B and C. Interestingly, by relaxing the structure of direct interdependencies in HIFF-II we create *inter-level interdependencies* (explained later in subsection D).

B. Number of distinct fitness values

Table 3 displays enumerations of the HIFF, HIFF-II and HXOR-II fitness functions for N=4 with differences between HIFF-II and HIFF fitness values highlighted. The HXOR-II function is the same as HIFF-II except points are rewarded for inequality. A comparison of the plots for HIFF and HIFF-II in Fig.5 shows HIFF-II has more points with different aggregate fitness values than HIFF. HIFF has 4 distinct aggregate fitness values while HIFF-II has 5.

TABLE 3

		HXOR-II		HIFF-II			HIFF			
gnum	genotype	*f*	*hd*	*L2*	*L1*	*f*	*L2*	*L1*	*f*	*hd*
0	0000	0.0	2	**1.0**	**2.0**	**3.0**	**1.0**	**2.0**	**3.0**	**0**
1	0001	1.5	1	0.5	1.0	1.5	0.5	1.0	1.5	1
2	0010	1.5	1	0.5	1.0	1.5	0.5	1.0	1.5	1
3	0011	1.0	2	0.0	**2.0**	2.0	0.0	**2.0**	2.0	2
4	0100	1.5	1	0.5	1.0	1.5	0.5	1.0	1.5	1
5	0101	2.0	2	**1.0**	0.0	1.0	0.5	0.0	0.5	2
6	0110	**3.0**	**0**	0.0	0.0	0.0	0.5	0.0	0.5	2
7	0111	1.5	1	0.5	1.0	1.5	0.5	1.0	1.5	1
8	1000	1.5	1	0.5	1.0	1.5	0.5	1.0	1.5	1
9	1001	**3.0**	**0**	0.0	0.0	0.0	0.5	0.0	0.5	2
10	1010	2.0	2	**1.0**	0.0	1.0	0.5	0.0	0.5	2
11	1011	1.5	1	0.5	1.0	1.5	0.5	1.0	1.5	1
12	1100	1.0	2	0.0	**2.0**	2.0	0.0	**2.0**	2.0	2
13	1101	1.5	1	0.5	1.0	1.5	0.5	1.0	1.5	1
14	1110	1.5	1	0.5	1.0	1.5	0.5	1.0	1.5	1
15	1111	0.0	2	**1.0**	**2.0**	**3.0**	**1.0**	**2.0**	**3.0**	**0**

gnum is genotype number, the integer equivalent of the binary string; $L\lambda$ is the fitness for level λ; *f* is aggregate fitness; *hd* is Hamming distance to a nearest global optimum. Optimal fitness values are bolded; discrepancies between the HIFF function and our HIFF-II function are highlighted.

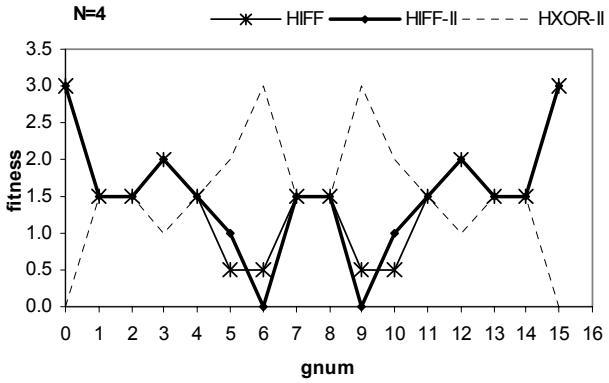

Fig. 5 HIFF, HIFF-II and HXOR-II fitness functions for N=4.

C. Number of competing optimal solutions per level

Table 3 reveals that the number of genotypes which can produce an optimal solution for a level is the same for all levels in HIFF-II. But this is not the case with HIFF. For HIFF-II, real 1.0, the optimal fitness value for level 2, occurs 4 times in column L2 and real 2.0, the optimal fitness value for level 1, also occurs 4 times in column L1. In contrast, 1.0 appears twice in column L2 and 2.0 appears 4 times in column L1, for HIFF. The optimal solutions per level for HIFF and HIFF-II are juxtaposed in Tables 4 and 5.

TABLE 4
SETS OF PER-LEVEL OPTIMAL GENOTYPES FOR N=4

Level	Optimal level fitness	HIFF		HIFF-II	
		Cardinality	Optimal genotypes	Cardinality	Optimal genotypes
2	1	2	{0000, 1111}	4	{0000, 0101, 1010, 1111}
1	2	4	{0000, 0011, 1100, 1111}	4	{0000, 0011, 1100, 1111}

TABLE 5
SETS OF PER-LEVEL OPTIMAL GENOTYPES FOR N=8

Level (λ)	Optimal level fitness	HIFF		HIFF-II	
		Size ($2^{N/2^{\lambda}}$)	Optimal genotypes	Size ($2^{N/2}$)	Optimal genotypes
3	1	2	{0000 0000, 1111 1111}	16	{0000 0000, 0001 0001, 0010 0010, 0011 0011, 0100 0100, 0101 0101, 0110 0110, 0111 0111, 1000 1000, 1001 1001, 1010 1010, 1011 1011, 1100 1100, 1101 1101, 1110 1110, 1111 1111}
2	2	4	{0000 0000, 0000 1111, 1111, 0000, 1111 1111}	16	{0000 0000, 0000 0101, 0000 1010, 0000 1111, 0101 0000, 0101 0101, 0101 1010, 0101 1111, 1010 0000, 1010 0101, 1010 1010, 1010 1111, 1111 0000, 1111 0101, 1111 1010, 1111 1111}
1	4	16	{0000 0000, 0000 0011, 0000 1100, 0000 1111, 0011 0000, 0011 0011, 0011 1100, 0011 1111, 1100 0000, 1100 0011, 1100 1100, 1100 1111, 1111 0000, 1111 0011, 1111 1100, 1111 1111}	16	{0000 0000, 0000 0011, 0000 1100, 0000 1111, 0011 0000, 0011 0011, 0011 1100, 0011 1111, 1100 0000, 1100 0011, 1100 1100, 1100 1111, 1111 0000, 1111 0011, 1111 1100, 1111 1111}

D. Inter-level interdependencies

Having a larger number of optimal solutions at every level of the hierarchy does not make HIFF-II an easier problem than HIFF because the sets of per-level optimal solutions are distinct from each other. This creates more competing schema per level and thwarts attempts to evolve a globally optimal solution that focus adaptation efforts on any one level. Unlike HIFF, an optimal solution for level λ+1 in a HIFF-II problem need not be an optimal solution for level λ, and vice versa. In this sense, the levels in HIFF-II are inter-dependent. There is dependency between levels in HIFF, but this dependency is one way. An optimal solution at a lower level need not translate to an optimal solution at a higher level in HIFF, but an optimal solution at a higher level does translate to an optimal solution at a lower level.

Table 5 lists the sets of optimal genotypes for each level in a problem with 8 variables. In HIFF, the set of optimal solutions for level λ+1 is a proper subset of the set of optimal solutions for level λ. In HIFF-II, the relationship is one of intersection and not subset. The two globally optimal genotypes are found in the intersection of the sets of per-level optimal genotypes. At every level, the HIFF set of optimal genotypes is a proper subset of the HIFF-II set of optimal genotypes. Therefore the intuition of the Building-Block Hypothesis [5, 9] is also preserved in HIFF-II.

E. Fitness Correlation

Fitness Distance Correlation (FDC) [10] is negative for straightforward problems and positive on deceptive problems. As a rule of thumb, search problems with FDC between -1.5 and 1.5 are difficult. In this section, FDC measures the correlation between per-level fitness values and Hamming distance to a closest global optimum. In Table 6 we see FDC values for HIFF increase at lower levels indicating that search is more difficult at lower levels than at higher levels. Previously, we exploited this feature of HIFF to accrete optimal HIFF solutions with a RMHC algorithm [2]. The RMHC algorithm was successful when it focused on optimizing higher level fitness values. However, the FDC values for HIFF-II are uniform across levels and this intuitively is in agreement with statements made in the previous subsection about the presence of inter-level interdependencies in HIFF-II. These summary statistics also apply to HXOR-II.

TABLE 6
FITNESS DISTANCE CORRELATION, N=8.

Level	HIFF	HIFF-II
3	-0.6972	-0.3486
2	-0.4930	-0.3486
1	-0.3486	-0.3486
Aggregate	-0.5711	-0.5325

IV. ALTRUISTIC AND NON-ALTRUISTIC SELECTION SCHEMES

A selection scheme is used in an evolutionary algorithm to decide whether to keep a genotype or to replace it with its variant. To make this decision, the selection scheme in the original RMHC algorithm compares the (aggregate) fitness of two genotypes. In our enhanced version of the RMHC algorithm, the selection scheme does a pair-wise comparison of the elements of two phenotypes in some order specified by the *sieve* component of the selection scheme.

Suppose the two competing phenotypes are $p_1 = \langle a, b, c \rangle$ and its variant $p_2 = \langle x, y, z \rangle$, and the selection scheme uses the following sieve: $\langle 1, 3, 2 \rangle$. Then the pair (c, z) is compared first because they are the fitness values for level 1, followed by (a, x), the fitness values for level 3 and lastly (b, y).

In a non-altruistic selection scheme, the first genotype which confers an advantage is selected. So in a maximizing problem, the genotype of p_2 is selected if one of $(z > c)$ or $(z = c$ and $x > a)$ or $(z = c$ and $x = a$ and $y \geq b)$ is true. If p_1 and p_2 are identical, the genotype of p_2 gets selected.

Under an altruistic selection scheme, a genotype is selected over its competitor only if it confers advantage on all levels. In other words, no level acts selfishly to further its progress at the

TABLE 7
THE ALTRUISTIC AND NON-ALTRUISTIC SELECTION SCHEMES USE SIEVE: ⟨3, 2, 1⟩.

gnum	Genotype	HIFF-II Phenotype λ=3	λ=2	λ=1	*f*	Selection (aggregate)	Selection (non-altruistic)	Selection (altruistic)
2 (parent)	0000 0010	0.75	1.5	3	5.25	3	2	2
3 (offspring)	0000 0011	0.5	1	4	5.5			
56 (parent)	0011 1000	0.25	0.5	3	3.75	56	57	56
57 (offspring)	0011 1001	0.5	0	2	2.5			
185 (parent)	1011 1001	0.75	0.5	1	2.25	186	186	186
186 (offspring)	1011 1010	0.75	1.5	1	3.25			

expense of other levels. In a maximizing problem, the genotype of p_2 is selected if ($z \geq c$ and $x \geq a$ and $y \geq b$) is true. If p_1 and p_2 are identical, the genotype of p_2 gets selected.

The order in which elements of a phenotype is compared does not matter under an altruistic selection scheme. This is not the case under a non-altruistic selection scheme. The altruistic selection scheme is distinct from comparing aggregate fitness values because in both HIFF and HIFF-II, aggregate fitness can improve at the cost of some per-level fitness value. Examples in Table 7 illustrate the differences among the three selection schemes mentioned in this section.

V. EXPERIMENTS

The hypotheses to be tested are:

1. HIFF-II is difficult for a traditional RMHC algorithm (RMHC1) but is not difficult for an enhanced RMHC algorithm that uses phenotypes and an altruistic selection scheme (RMHC3).

2. HIFF-II is difficult for an enhanced RMHC algorithm that uses phenotype but does not use an altruistic selection scheme (RMHC2). This holds even when macro-mutation is used, when sieves are varied over time (Multi-population) or when sieves are not varied over time but are randomly generated (Parallel-model). In other words, HIFF-II is difficult to solve with a RMHC algorithm that does not use an altruistic selection scheme.

3. HIFF-II is easier to solve with a genetic algorithm than a RMHC. We compare the performance of a genetic algorithm (GA-DC) with the performances of other RMHC algorithms including multi-population and parallel-model, but excluding RMHC3. We consider RMHC3 an ideal algorithm for HIFF-II. As such we compare RMHC3 with an "ideal" genetic algorithm (GIGA). We regard GIGA as ideal because population size can be small and yet population diversity is maintained throughout a run.

Performance of algorithms is measured in terms of the number of times a globally optimal solution is found in a set of runs and the number of evaluations used to evolve a globally optimal solution.

A. Algorithms

1. RMHC1.

(i) Generate a random bit string of length N.

(ii) Generate a random integer from [1, $P_m \times N$] for k.

(iii) Make a copy* of the current (parent) bit string. Select k bits at random from the copy bit string to mutate. Mutation is by random assignment of a 1 or a 0.

(iv) Replace the current bit string with the variant bit string if aggregate fitness of the variant (offspring) bit string is greater than or equal to aggregate fitness of the current bit string.

(v) If optimal string has not been found and maximum evaluations not reached, go to step (ii).

2. RMHC2

This is the method we proposed previously for HIFF [2]. It is the same as RMHC1 except that, in step (iv) phenotypes are compared and a non-altruistic selection scheme is used.

In this paper, we ran RMHC2 with two types of sieves, two kinds of mutation operator and two mutation rates. With a high-low sieve, comparisons of phenotype elements are made in descending order starting with the highest level. The high-low sieve for N=128 is ⟨7, 6, 5, 4, 3, 2, 1⟩. A low-high sieve is the exact opposite of a high-low sieve; phenotype elements are compared in ascending order starting with the lowest level. If macro-mutation is used in step (iii), the *k* consecutive bits are mutated starting from a random locus wrapping around to the start of the genotype if necessary.

3. RMHC3

The same as RMHC2 except that, in step (iv) phenotypes are compared with an altruistic selection scheme, as explained in section IV.

4. Multi-population

Reference [2] describes this algorithm in detail. Briefly, subpopulations of genotypes are placed in a two-dimensional grid with periodic boundaries. Genotypes of a subpopulation do not interact with each other and subpopulations have minimal interaction with each other. There is no exchange of genes between genotypes in this model.

Genotypes in a cell are evolved using RMHC2 and the current sieve in the cell. Sieves are generated at random initially. The number of changes made to genotypes of a subpopulation is monitored. After a specified number of generations, subpopulations compare their activity level (number of changes) with that of their neighbours. The most

* When P_m is low and N is large, it is more efficient implementation wise to remember the state of the k bits before mutation and restore the bits accordingly than to copy entire bit strings.

active subpopulation replaces the least active subpopulation in a Moore neighbourhood. During this extinction-recolonization event, the sieve in the recolonized cell is replaced by a variant of the sieve in the colonizer cell.

5. Parallel-model
The same as multi-population except there is no colonization or extinction of subpopulations and no mutation of sieves. A subpopulation adapts to the conditions imposed by a single fixed sieve, randomly generated at the start, for the entire run.

6. GA-DC
We use the steady-state GA-DC algorithm given in [1, p.168].
(i) Create population of random bit strings and evaluate population.
(ii) Select two random genotypes, p1 and p2.
(iii) With probability P_c, recombine p1 and p2. Evaluate pair of offspring genotypes c1 and c2.
(iv) Calculate Hamming distance *H*, between the following pairs of genotypes (p1, c1), (p1, c2), (p2, c2) and (p2, c1).
If H(p1, c1) + H(p2, c2) < H(p1, c2) + H(p2, c1), then pair p1 with c1 and p2 with c2 else pair p1 with c2 and p2 with c1.
(v) For each pair, if the offspring genotype is fitter than the parent genotype, replace the parent genotype with the offspring genotype, else keep the parent genotype and discard the offspring genotype.
(vi) If optimal solution has not been found and maximum evaluations is not reached, go to step (ii).

7. GIGA
(i) Create population of random bit strings. Evaluate population and sort population by aggregate fitness.
(ii) Starting from the fittest end of the sorted population, pair up consecutive genotypes so parent genotypes are close in aggregate fitness value. We cycle back to the fittest end when the least fit end is reached.
(iii) Produce offspring genotypes from parent genotypes with two-point crossover.
(iv) Replace the pair of parent genotypes with the pair of offspring genotypes if the fitter offspring is fitter than the fitter parent (elitism). The population is kept sorted and freshly minted offspring may participate in mating events immediately.

B. Parameters

Unless stated otherwise, the parameter values in Table 8 were used in the experiments. Runs were performed with a different random number seed each time.

C. Results

Table 9 reports the performance of the seven algorithms described in subsection A on the HIFF-II problem. From these results, we draw the following conclusions:

(i) Hypothesis 1 is confirmed. RMHC3 significantly performs better than RMHC1 on HIFF-II. On the 128 variable HIFF-II problem 76% of RMHC3 runs found an optimum. None of the 30 RMHC1 runs found an optimum within 3 million evaluations.

(ii) Hypothesis 2 is confirmed. RMHC3 was not outperformed by RMHC2, multi-population or the parallel-model. None of the RMHC2 runs found a solution to HIFF-II. None of the parallel-model runs were successful. One third of the multi-population runs were successful but they took over a million evaluations on average to find a solution while RMHC3 took considerably fewer evaluations on average.

The RMHC2 algorithm, which prioritizes progress at higher levels over lower levels, has the lowest average best final fitness. In marked contrast, RMHC2 was very successful on the HIFF problem: all runs succeeded and the average evaluation was 2245 with a standard error of 660 [2]. This difference in RMHC2 performance highlights the fact that HIFF-II has a different kind of inter-level dependency from HIFF, as explained in section III.

Table 10 shows the average per-level fitness at the end of a sample of unsuccessful runs. The biases of each algorithm are revealed. RMHC1, RMHC2-A and RMHC2-B are biased towards optimization of the lower level modules while RMHC2 is biased towards the higher level modules. On the other hand, RMHC3 is even handed in its dealings with each level. These findings are not unexpected.

TABLE 8

Application	Parameter	Default Value
All	Maximum evaluations per run	1,000,000
	Problem size, N	128
All RMHC-s	Mutation rate, P_m	0.0625
Multi-population Parallel-model	Lattice dimensions	4×4
	Population size per deme	5
	Number of generations (for multi-population only)	100
	Mutation rate, P_m	0.0625
GA-DC GIGA	Population size for GA-DC	2000
	Population size for GIGA	50
	Recombination type	2 point
	Recombination rate, P_c	1.0
	Mutation rate, P_m	0.0

(iii) Hypothesis 3 is confirmed. The GA-DC runs were significantly more successful than the RMHC2 runs. The GA-DC runs found an optimum at least 90% of the time. None of the RMHC2 runs found an optimum. Further, the successful GA-DC runs used significantly fewer evaluations than the successful multi-population runs.

The GIGA runs were more successful than the RMHC3 runs. On the 128-variable problem, the success rate is 100% for the GIGA algorithm, and 76% for the RMHC3 algorithm. However, there is no significant difference in the average number of evaluations used in successful runs.

On the 256-variable problem, the disparity is more evident not only in the number of times an optimal solution was found but also in the efficiency of successful runs. The *t* test was used to test significance. The 1-tailed probability that there is

TABLE 9
SUMMARY OF RESULTS FOR EXPERIMENTS WITH **N=128**. AVERAGE BEST FINAL FITNESS IS RREPORTED FOR UNSUCCESSFUL RUNS. AVERAGE NUMBER OF EVALUATION USED IS REPORTEDFOR SUCCESSFUL RUNS.

		HIFF-II		
Method	**Character**	**Times found**	**Avg. Best Final Fitness (std. dev.)**	**Avg. Evaluations (std. dev.)**
RMHC1	compare aggregate fitness, maximum evaluations = 3,000,000	0/30 (0 %)	115.92 (2.34)	-
RMHC2	compare phenotypes, selfish selection scheme, high-low sieve	0/10 (0 %)	66.55 (7.50)	-
RMHC2-A	compare phenotypes, selfish selection scheme, low-high sieve	0/10 (0 %)	114.14 (2.52)	-
RMHC2-B	compare phenotypes, selfish selection scheme, low-high sieve, macro-mutation ($P_m = 0.5$)	0/10 (0 %)	123.65 (0.62)	-
Multi-population	compare phenotypes, selfish selection scheme, variable sieve, maximum evaluations = 3,000,000.	5/15 (33 %)	104.18 (9.41)	1,659,579 (983,627)
Parallel-model	compare phenotypes, selfish selection scheme, fixed random sieve, maximum evaluations = 3,000,000.	0/10 (0 %)	109.42 (4.22)	-
GA-DC1	compare aggregate fitness, one-point crossover, no mutation, $P_c = 0.7$ This parameter configuration is used in [1].	27/30 (90 %)	126.00 (0.00)	252,167 (171,326)
GA-DC2	compare aggregate fitness, two-point crossover, no mutation This parameter configuration is used to compare with GIGA.	30/30 (100 %)	-	166,370 (50,909)
RMHC3	compare phenotypes, altruistic selection scheme	23/30 (76 %)	118.45 (5.10)	60,415 (38,689)
GIGA	compare aggregate fitness, two-point crossover, no mutation, gene-invariant	30/30 (100 %)	-	63,084 (20,183)
RMHC3	compare phenotypes, altruistic selection scheme, N=256	22/30 (73 %)	233.97 (25.58)	399,281 (208,316)
GIGA	compare aggregate fitness, two-point crossover, no mutation, gene-invariant, N=256	30/30 (100 %)	-	245,076 (82,916)

Notes:
(i) We did not test more GA variations because our objective is not to identify the "best" GA for HIFF-II, but to show that a GA can solve HIFF-II more effectively than a RMHC that is "ideal" for HIFF-II (RMHC3) while still keeping mutations random.
(ii) In keeping with [1] and our preliminary tests which favoured random bit assignment, we did not use bit-flip mutation. However, in response to a reviewer's comment, we ran RMHC3 with the suggested configuration ($P_m = 1.0$ and mutation by bit-flip) and obtained the following result for N=128: only 10 out of 30 runs were successful after 1million evaluations. The successful runs took on average 643,200 number of evaluations with a standard deviation of 229,130. Clearly, this performance is worse than that reported in Table 9. One reason for the poor performance is the high mutation rate. In our configuration, the effective mutation rate is lower than reported because our mutation method is random bit assignment which does not necessarily change a bit selected for mutation.
(iii) Doing 30 runs each for the RMHC2 experiments did not alter the rate of finding a global optimum. No RMHC2 run, of any flavour, succeeded.
(iv) In response to a reviewer's comment about the difference in signal (min - max), we reply: (a) The GAs we use in our experiments do not make use of fitness proportionate selection where scaling can be an issue; (b) RMHC2 does not work on discrete HIFF because the fitness landscape of the highest level is 2 needles in a haystack. We are working to address this.

TABLE 10
AVERAGE LEVEL FITNESS AT END OF UNSUCCESSFUL RUNS, N = 128.

	Level (λ)	**7**	**6**	**5**	**4**	**3**	**2**	**1**
Method	**Optimal fitness**	**1**	**2**	**4**	**8**	**16**	**32**	**64**
RMHC1 (15 runs)		0.49 (49 %)	1.07 (53 %)	2.35 (58 %)	4.83 (60 %)	11.8 (73 %)	32.0 (100 %)	64 (100 %)
RMHC2 (10 runs)		1.00 (100 %)	2.00 (100 %)	2.25 (56 %)	4.30 (53 %)	8.6 (53 %)	16.4 (51 %)	32 (50 %)
RMHC2-A (10 runs)		0.58 (58 %)	1.15 (57 %)	1.80 (45 %)	4.60 (57 %)	10.0 (62 %)	32.0 (100 %)	64 (100 %)
RMHC2-B (10 runs)		0.45 (45 %)	1.10 (55 %)	2.30 (57 %)	7.80 (97 %)	16.0 (100 %)	32.0 (100 %)	64 (100 %)
RMHC3 (7 runs)		0.93 (93 %)	1.87 (93 %)	3.64 (91 %)	7.57 (94 %)	14.8 (92 %)	29.5 (92 %)	60 (93 %)

no significant difference between the two averages (399,281 – 245,076) is 10% with 50 degrees of freedom.

VI. Conclusion

We introduced a new test problem, HIFF-II. The experiments we conducted confirm that (i) HIFF-II is difficult to solve with a traditional RMHC algorithm but can be solved with an enhanced RMHC if an altruistic selection scheme is used, and that (ii) genetic algorithms which maintain population diversity have an easier time evolving a solution to HIFF-II than RMHC algorithms of any flavour. Point (i) gives empirical evidence that HIFF-II is a different problem from HIFF. This difference lies in the type of inter-level dependency. In HIFF, inter-level dependency is one-way from the bottom-up, that is lower level modules are dependent on higher level modules to find their "true" optimal solution. But in HIFF-II, the inter-level dependency is bi-directional, bottom-up and top-down. Higher level modules depend on lower level modules for their "true" optimum and vice versa. Thus in HIFF-II levels are interdependent. It is this inter-level interdependency, coupled with modular interdependency at each level that defeats the RMHC algorithms.

Acknowledgement

Thanks to Dr. P. Grogono and the anonymous reviewers for their helpful comments. I am funded by NSERC and the Faculty of Engineering and Computer Science, Concordia University.

References

1 R. A. Watson, "Compositional Evolution", 2006, The MIT Press.

2 S. Khor, "Rethinking the adaptive capability of accretive evolution on hierarchically consistent problems," IEEE Artificial Life Symposium, 2007.

3 S. Forrest and M. Mitchell, "Relative building-block fitness and the building-block hypothesis," in D. Whitley (editor) Foundations of Genetic Algorithms (FOGA) vol. 2, 1993, Morgan Kaufmann.

4 R. A. Watson, G. S. Hornby, and J. B. Pollack, "Modeling building-block interdependency," in A.E. Eiben, T. Bäck, M. Schoenauer and H.-P. Schweffel (editors) Parallel Problem Solving from Nature (PPSN) vol. V, 1998, pp. 97 – 106, Springer, Berlin.

5 J. H. Holland, "Adaptation in natural and artificial systems," 1992, The MIT Press.

6 W. M. Spears, "Crossover or mutation?" in L. D. Whitley (editor), Foundations of Genetic Algorithms (FOGA) vol. 2, 1992, Morgan Kaufmann.

7 J. C. Culberson, "Mutation-crossover isomorphisms and the construction of discriminating functions," Evolutionary Computation vol. 2, 1994, pp. 279-311.

8 S. Mahfoud, "Niching methods for genetic algorithms," Ph.D. Dissertation, 1995, University of Illinois.

9 D. E. Goldberg, "Genetic algorithms in search, optimization and machine learning," 1989, Addison-Wesley Publishing Company.

10 T. Jones and S. Forrest, "Fitness distance correlation as a measure of problem difficulty for genetic algorithms," in L. Eshelman (editor), 6th Interrnational Conference on Genetic Algorithms (ICGA), 1995, pp. 184 – 192, Morgan Kaufmann.