The metapopulation genetic algorithm: An efficient solution for the problem of large phylogeny estimation

Alan R. Lemmon[†] **and Michel C. Milinkovitch**[‡]

Laboratory of Evolutionary Genetics, Free University of Brussels (ULB), Cp 300, Institute of Molecular Biology and Medicine, Rue Jeener and Brachet 12, B-6041 Gosselies, Belgium

Edited by John C. Avise, University of Georgia, Athens, GA, and approved June 17, 2002 (received for review April 15, 2002)

Large phylogeny estimation is a combinatorial optimization problem that no future computer will ever be able to solve exactly in practical computing time. The difficulty of the problem is amplified by the need to use complex evolutionary models and large taxon samplings. Hence, many heuristic approaches have been developed, with varying degrees of success. Here, we report on a heuristic approach, the metapopulation genetic algorithm, involving several populations of trees that are forced to cooperate in the search for the optimal tree. Within each population, trees are subjected to evaluation, selection, and mutation events, which are directed by using inter-population consensus information. The method proves to be both very accurate and vastly faster than existing heuristics, such that data sets comprised of hundreds of taxa can be analyzed in practical computing times under complex models of maximum-likelihood evolution. Branch support values produced by the metapopulation genetic algorithm might closely approximate the posterior probabilities of the corresponding branches.

heuristics | evolutionary computation | maximum likelihood

Optimality criterion-based phylogeny inference is a notoriously difficult endeavor because the number of solutions increases explosively with the number of taxa. Indeed, the total number of possible unrooted, bifurcating tree topologies among T-terminal taxa is

$$B(T) = \prod_{i=3}^{T} (2i - 5) \text{ i.e., } \frac{(2T-5)!}{(T-3)!2^{T-3}},$$

corresponding to nearly 32 billion different trees for 14 taxa and 3×10^{84} trees (i.e., more than the number of atoms in the known universe) for 55 taxa. Phylogeny inference is a combinatorial optimization problem of nondeterministic polynomial (NP)-complete type (1, 2) because (*i*) no known algorithm can solve it in polynomial time, and (*ii*) demonstrating the existence of such an algorithm would imply that all NP problems have a polynomial time solution. As most mathematicians expect that no such algorithm exists, one is forced to admit that no future civilization will ever build a computer capable of solving the problem while guaranteeing that the optimal solution has been found.

Computation time has only recently become a central and widespread practical problem in phylogeny inference for two main reasons. First, major advances in molecular biology and biotechnology have caused a dramatic increase in the number of DNA sequences available, stimulating researchers to increase their taxon sampling when performing tree reconstruction. Incidentally, several authors (e.g., ref. 3) have suggested that the accuracy of phylogeny inference increases with increased taxon sampling. Second, many simulation studies in the last 10 years (e.g., ref. 4) have identified the maximum-likelihood (ML) criterion (5) as one of the best for phylogeny inference. Reasons for this include (*i*) statistical consistency (ML estimators tend to converge to true parameters values when the number of characters is increased), (*ii*) robustness (violations of the ML model assumptions have only a moderate impact on the tree inference accuracy), (*iii*) the ability to compare different trees within a statistical framework, and (*iv*) the ability of ML to make full use of the original character matrix. Unfortunately, the analytical power of ML has a cost: computation time. Hence, application of this model-based criterion makes the use of exact phylogeny inference methods impractical for more than a trivial number of taxa (about 12). Note that (*i*) by ML we mean its most widely implemented version, i.e., maximum average likelihood (which is itself a form of the more general maximum relative likelihood approach; ref. 6), and (*ii*) in addition to optimality criterion-based phylogeny inference being proven NP-complete because of the number of trees, it remains an open question whether maximum average likelihood can be calculated efficiently (i.e., in polynomial time) on a single tree.

As with many NP-complete problems, the only practical approach to large phylogeny inference is the use of heuristics, i.e., methods that find one or several solution(s) faster than exact searches while sacrificing a guarantee that the optimal solution will be found. Given the tremendous number of new questions in evolutionary biology that could be investigated through the use of larger taxon samplings, most researchers are ready to give up the quest for the absolute optimal tree provided that alternative heuristic methods yield optimal or near-optimal solutions with high probability. In response to this trend, much of the current research in computational phylogenetics concentrates on the development of more efficient heuristic approaches.

Many existing heuristic methods are handicapped with the need to optimize model parameters (such as branch lengths) on each tree examined, a procedure that significantly slows computation time. One such heuristic, which is implemented in the most widely used inference packages [PAUP* (7), PHYLIP (8), etc.], is based on hill climbing: an initial tree [most commonly obtained by using the stepwise addition (StepAdd) algorithm (9)] is subjected to a topological rearrangement known as branch swapping (9), followed by an optimization of branch lengths and other model parameters. If the new tree has a better score than the starting tree, it is kept and used as the new starting tree; otherwise the proposed tree is rejected. The procedure is repeated in a systematic fashion, until no swapping on the current best tree can improve the score. Because of the need to optimize parameters at every step (intra-step optimization), simple branch-swapping methods remain computationally im-

This paper was submitted directly (Track II) to the PNAS office.

Abbreviations: ML, maximum likelihood; GA, genetic algorithm; metaGA, metapopulation GA; HKY, Hasegawa-Kishino-Yano; JC, Jukes-Cantor; CP, consensus pruning; SPR, subtree pruning-regrafting; NNI, nearest-neighbor interchange; TBR, tree bisection-reconnection; StepAdd, stepwise addition.

[†] Present address: Section of Integrative Biology PAT CO930, University of Texas, Austin, TX 78712.

[‡] To whom reprint requests should be addressed. E-mail: mcmilink@ulb.ac.be.

www.pnas.org/cgi/doi/10.1073/pnas.162224399

practical for more than 25–50 taxa, depending on the complexity of the ML model.

Two classes of solutions have been developed in response to the problems associated with optimizing parameters on large trees. The first of these classes includes solutions that partition the large problem of tree reconstruction into many small subproblems whose solutions are then combined into a consensus global solution. For example, the quartet puzzling method (10) works by reconstructing the best ML tree of each possible quartet of taxa, then combining (i.e., puzzling) the quartet trees to construct an overall T-taxon tree. Because the puzzling procedure depends on the order with which the taxa are sequentially inserted, it is typically repeated *n* times (with randomization of the T-taxa input order), and a majority-rule consensus tree is computed from the *n* T-taxa trees. The advantage of quartet puzzling is that it avoids numerous optimizations on large trees, optimizing instead numerous small trees, which is computationally much simpler. Despite the initial appeal, the method has proven to be only moderately more accurate (10) than neighbor joining (11), a pairwise-distance based method that requires significantly less computation time.

The second class is comprised of stochastic heuristics that avoid optimization of numerous trees entirely. Instead, they incorporate methods that allow branch lengths and other model parameters to be optimized as the search proceeds, taking an inter-step optimization strategy. Stochastic simulated annealing (SSA; ref. 12), for example, is based on the simple branch-swapping algorithm described above, but incorporates a method of perturbing branch lengths at each iteration instead of requiring optimization of each potential solution. SSA avoids local optima by accepting changes that decrease the likelihood of the tree with a probability inversely proportional to the reduction in likelihood. Another increasingly popular approach is the Bayesian method (13) based on the Markov chain Monte Carlo (MCMC) algorithm. MCMC-based methods also benefit from avoiding intra-step optimization, although they have a slightly different aim: sampling the distribution of tree space instead of only finding optimal trees.

Finally, Matsuda (14), Lewis (15), and Katoh *et al.* (16) have recently applied the genetic algorithm (GA), a type of evolutionary computation method (17), to phylogeny inference. GAs implement a set of operators that mimic processes of biological evolution such as mutation, recombination, selection, and reproduction. After an initial step of generating a population, the individuals (specific solutions) within that population are (*i*) subjected to mutation and/or recombination and (*ii*) allowed to reproduce with a probability that is a function of their relative fitness value. In the case of a phylogenetic inference problem, individuals are typically composed of topologies and model parameters (e.g., branch lengths, the transition/transversion ratio, rate heterogeneity parameters, etc.) that need to be optimized. A mutation is a stochastic alteration of one component of the individual (e.g., a topological rearrangement, a change in one branch length, or a random modification of a model parameter), and the fitness of an individual is a function of the score for that tree. Because selection retains the changes that improve the value of the optimality function, the mean score of the population of trees improves over time, i.e., across generations.

We report here on a GA, nicknamed the metapopulation GA (metaGA), that vastly improves the speed and efficiency with which ML trees are found (such that nucleotide sequence data sets incorporating hundreds of taxa can be analyzed in practical computing times) and yields a probability index for each branch. We incorporated the metaGA procedure into a computer program, METAPIGA (phylogeny inference using the metaGA), whose advantages are: (*i*) rapid exploration of search space and identification of optimal trees, (*ii*) identification of multiple optima within a single search, (*iii*) flexible GA structure that allows fine control over both speed and accuracy, (*iv*) production of branch probability indices, and (*v*) a user-friendly interface. Using simulated and real data sets, we investigated the effect of major search parameters on the speed and accuracy of the metaGA procedure. Data sets of 500 taxa and 3,000 nt can be analyzed on a regular computer (1.7-Gh Pentium IV) in 10 h under the Hasegawa-Kishino-Yano (HKY) + rate heterogeneity model. METAPIGA (version 1.0) is written in the cross-platform-compatible (Windows, Macintosh, Unix/Linux) Java programming language and implements a graphical interface for importing/exporting data, specifying a ML model, monitoring search progress, visualizing trees, and performing statistical analyses of trees generated by the GA. METAPIGA (and a supporting user's manual) is distributed at http://dbm.ulb.ac.be/ueg.

Materials and Methods

ML Models and the Basic GA Cycle of Evaluation-Selection-Mutation. METAPIGA implements the HKY nucleotide substitution model (18), as well as the nested Jukes-Cantor (JC) (19), K2P (20), and F81 (5) models. More general models of nucleotide substitution (e.g., general time-reversible) will be incorporated into future versions of METAPIGA. The transition/transversion ratio is optimized periodically during the search with a frequency specified by the user. Equilibrium base frequencies remain constant throughout the search and are either specified by the user or estimated from the data set. Rate heterogeneity is incorporated through a method similar to that described by Van de Peer *et al.* (21), which is based on corrected distance estimation. Use of this method allows estimation of rate heterogeneity parameters once, before the start of the search. The GA cycle of evaluation-selection-mutation implemented in METAPIGA is described in the supplemental *Materials and Methods* and *Results and Discussion*, which are published as supporting information on the PNAS web site, www.pnas.org.

The MetaGA. *The essence of the procedure.* We designed a family of heuristic search strategies, the metaGAs. This approach relies on the coexistence of two or more populations interacting in a metapopulation setting. Both the number of populations and the

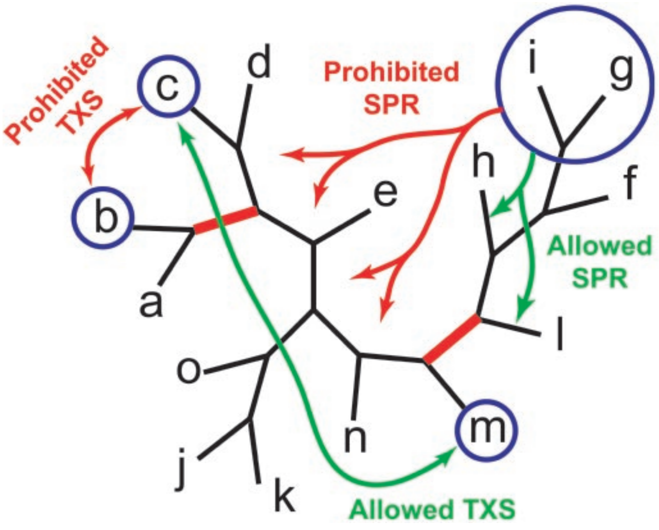

Fig. 1. The principle of CP. Before a tree is subjected to mutation, its topology is compared with those of the best tree(s) from other populations; the consensus branches (indicated in bold red) define the partitions that can (green arrows) and cannot (red arrows) be affected by topological mutations; i.e., any operation moving a taxon across a consensus branch is prohibited. The method used to define a consensus branch depends on the CP option chosen by the user (see text for details). TXS, taxa swap. See supporting *Materials and Methods* for details.

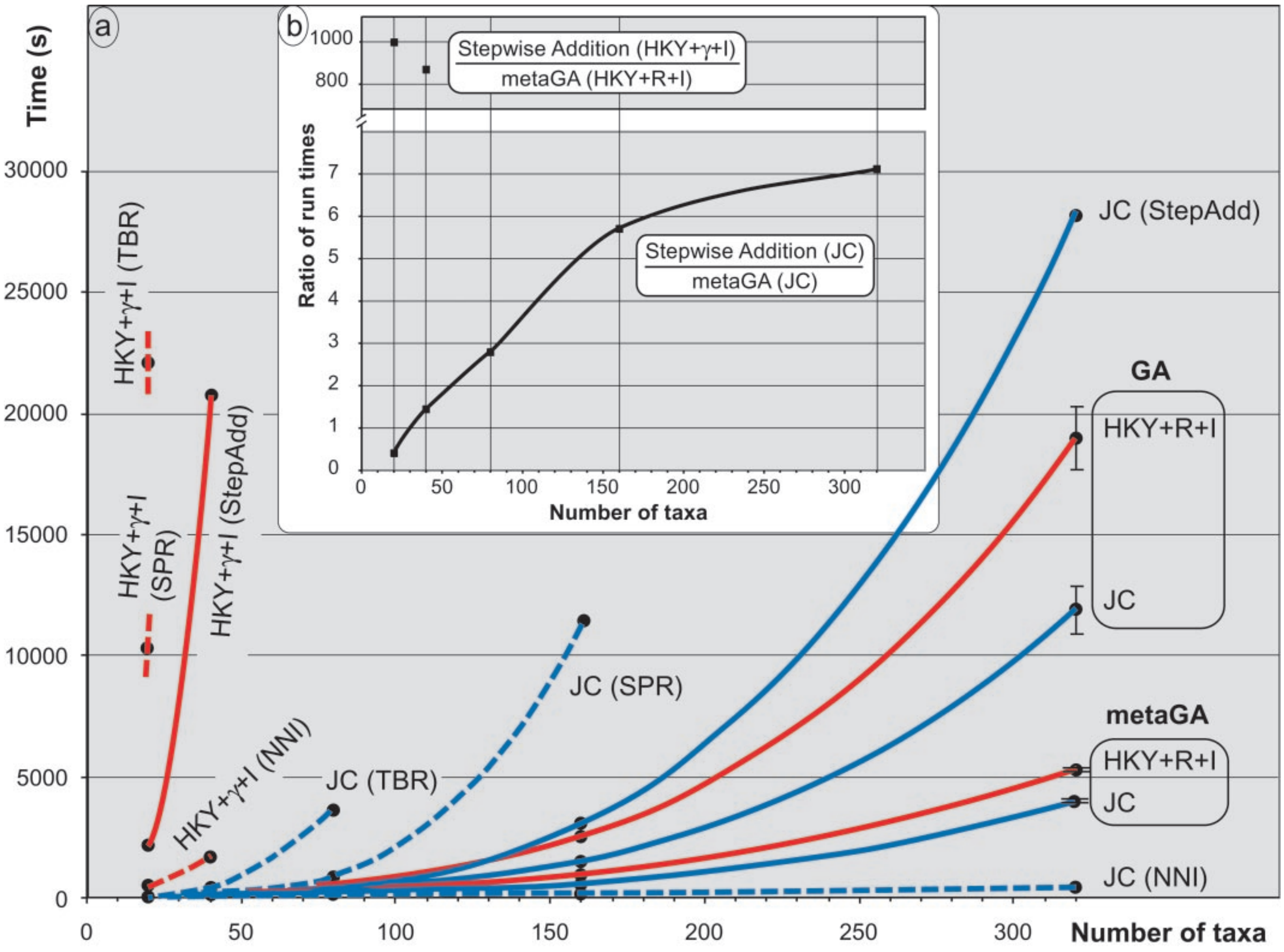

Fig. 2. (*a*) Relative run times vs. number of taxa for the GA (one population), metaGA (with strict group consensus among four populations) and StepAdd algorithm (9) as well as for a single round of NNI, SPR, and TBR swapping (dotted lines), under the JC (blue) and HKY + rate heterogeneity (red) ML models. Standard errors (among 10 runs) are indicated for the GA and metaGA only. (*b*) Ratio between StepAdd and average metaGA run times (vs. number of taxa) under the JC (*Lower*) and HKY + rate heterogeneity (*Upper*) ML models.

number of individuals per population are specified by the user. As the metaGA involves several parallel searches, a large amount of inter-population variation can be maintained, even when each population is subjected to strong selection pressures. However, the spirit of the metaGA is that the populations are not fully independent as they are forced to cooperate in the search for the optimal tree. Within each population, trees are subjected to evaluation, selection, and mutation events as they would be in a single-population GA (see supporting *Materials and Methods*), but all topological operators are guided through inter-population comparisons. The metaGA is based on the assumption that these comparisons allow the identification of partitions that are correct (and should not be modified) and regions that still need to be resolved. Communication among the populations is defined and controlled through a principle we named consensus pruning (CP), i.e., consensus information gathered from comparison of the best trees across populations is used to identify the portions of each tree that can be subjected to mutations (Fig. 1). The concept of CP allows the elaboration of many specific inter-population communication procedures of which we chose to implement six: random, ring, alternate ring, strict group consensus, majority group consensus, and probability group consensus. When CP is set to random during a metaGA search, each of the P populations is picked in turn and randomly paired with one of the remaining P-1 populations. The consensus branches between the two trees define the partitions that cannot be affected by topological mutations. For example, if a consensus branch defines a partition between groups I and II, no taxa swap between a taxon in group I and a taxon in group II is allowed, whereas topological changes within group I and within group II are allowed (Fig. 1). Under the ring option, populations are ordered in a circle and each population *p* is paired with the next population in line ($p + 1$), with the *P*th population paired with population 1. Alternate ring is the same as ring except that the direction of pairing switches every *G* generations (*G* defined by the user). Under strict group consensus and majority group consensus, the partitions that cannot be affected by topological mutations are defined as the branches exhibited by the strict consensus and the 50% majority consensus, respectively, among the best trees from all populations. Finally, under probability group consensus, each partition is enforced with a probability proportional to the percentage of populations that agree on that partition. Variations of the metapopulation concept, such as local extinction and recolonization, may also prove to be effective, although we have yet to explore such approaches. To our knowledge, the single heuristic that could be considered related

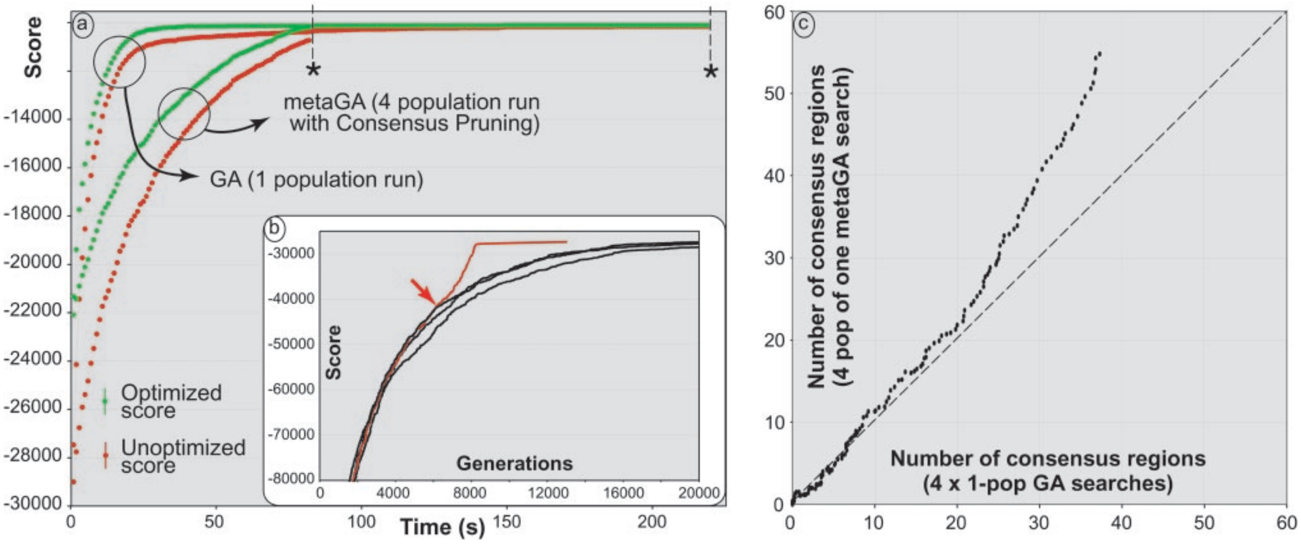

Fig. 3. (*a*) Unoptimized (red dots) and optimized (green dots) score vs. time for GA (one population) and metaGA (strict CP with four populations of four individuals each) runs (JC model, 80 taxa). The asterisks indicate when the stopping rule has been reached. (*b*) Score of best tree (320 taxa) for each of four populations run in parallel (no CP). At the time indicated by the red arrow, one population (red) was allowed to use the consensus information from the three other populations (black). This process demonstrates that scores increase faster when CP is enabled. (*c*) Number of consensus partitions among four independent 80-taxa populations running under the GA (JC model) vs. number of consensus partitions among four 80-taxa populations interacting through CP during a metaGA run.

to the metaGA concept is the recently developed hitchhiking strategy (21) where a set of hitcher trees are allowed (when possible) to experience the same perturbation as a driver tree. Hitchhiking is based on the assumption that identical perturbations could gradually drive different temporary solutions toward the same global optimum, whereas the metaGA assumes that topological consensus among different temporary solutions allows the identification of unacceptable perturbations. In hitchhiking, one solution partially drives the others, whereas in the metaGA all solutions constantly check each other.

Stopping rule. As the overall number of consensus partitions increases with search time, the metaGA procedure provides a convenient stopping rule: the search stops when (*i*) the best trees of all populations have identical topologies, or (*ii*) when all topological changes allowed by the latest consensus information have been attempted on the best trees of all populations and these changes did not improve their scores. Given the high percentage of partitions that are fixed toward the end of the search, swapping to completion is swift even on very large trees. This stopping condition option is called GA decides. Other stopping rules available in METAPIGA are: stop when the user hits the stop button (user decides option), stop when the number of generations predefined by the user has been reached (fixed-generation option), stop when no topological change has been accepted by the GA in the last *n* generations (topol improvement option; *n* defined by the user), or stop when a score is obtained that is better than that specified by the user (target score option).

After the stopping case has been reached, branch lengths are optimized by the Newton–Rhapson method (23). All parameters specific to the chosen nucleotide substitution model are also optimized at this point. Once optimized, the best tree from each population can be stored, viewed, and exported. We also provide in METAPIGA the option to keep in memory the *n* trees (*n* specified by the user) with the best scores from each population's history.

Results and Discussion

We first defined a set of default search settings (supporting *Materials and Methods*): one population of four individuals; selection option = improve, probability of branch length mutation = 0.04; probability of each branch swapping operator [subtree pruning-regrafting (SPR), nearest-neighbor interchange (NNI), taxa swap, subtree swap) = 0.24; probability of recombination = 0; dynamic operators disabled; CP disabled; starting trees = random; stopping rule = GA decides; JC model (i.e., the simplest ML model). We chose to use the JC model for testing the dynamics of the metaGA because it would allow for the most rapid searches, allowing us to test more parameters of the metaGA. All results reported below aim at systematically testing the effect, on both computing time and accuracy of topology inference, of varying one or several of these search parameters. These results are compared with the performances of algorithms implemented in two widely used software packages (7, 24): MRBAYES 2.01 and PAUP*4.b8. We are fully aware of the difficulties associated with comparing runtimes of algorithms implemented in different software packages. However, the reader cannot know the relative efficiency of the methods described above unless some comparisons are made. Two things are done to assure that the most appropriate/useful comparisons are made. First, all analyses (METAPIGA 1.0, MRBAYES 2.01, and PAUP*4.b8) were performed on the same computer running WINDOWS 2000 (single 1.7-Gh Pentium IV processor with 1 Gb of RAM). Second, we compare the efficiency of our method with methods that are most commonly used; hence, we chose to use default search settings. Analyses were performed on simulated data sets comprised of 1,000 nt and either 20, 40, 80, 160, or 320 taxa. All data sets were generated by using DNA-SIM, a program written by A.R.L. and based on a birth-and-death process (6, 25) (JC; speciation rate = 10^{-4}; extinction rate = 10^{-5}; mutation rates = 9.0×10^{-3}, 6.0×10^{-3}, 3.5×10^{-3}, 2.5×10^{-3}, 2.0×10^{-3} for 20, 40, 80, 160, and 320 taxa, respectively). We also tested the efficiency of our metaGA by using a 55-taxa *rbc*L data set of 1,314 nt (15).

GA (i.e., CP Disabled), MetaGA (i.e., CP Enabled), and Other Heuristics. Fig. 2 indicates, both for the JC and HKY + rate heterogeneity (four categories + invariable sites) models, that the time required by a GA search (one population run to completion) is much less than that required by StepAdd. The use of the

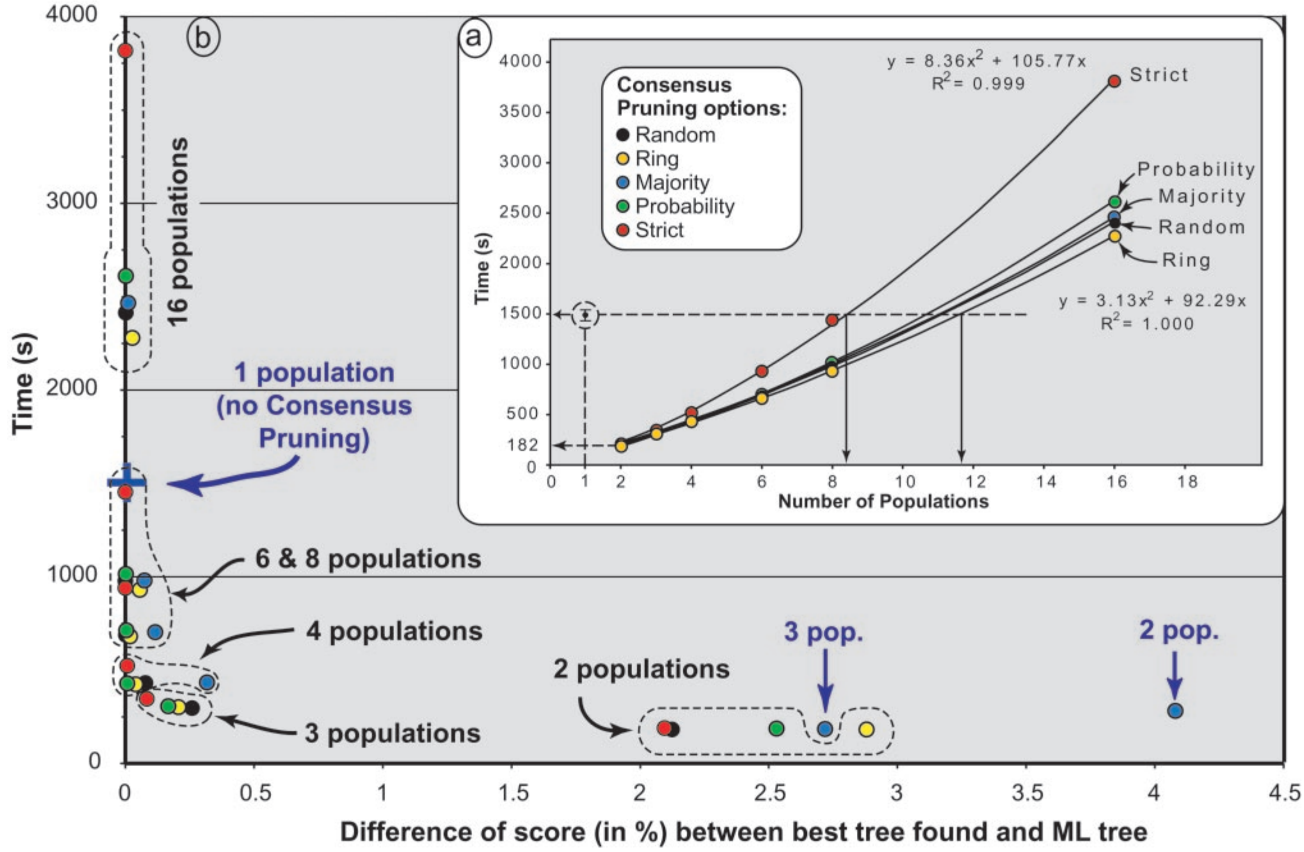

Fig. 4. (*a*) Run times (JC model, 160 taxa) for a one-population GA search (dotted circle; ± SE indicated) as well as for two-, three-, four-, six-, eight-, and 16-population metaGA searches under each of the CP options (10 runs each, SE too small for being visible). The vertical arrows delimit the range in number of populations for which a metaGA run (depending on the CP option) can take the same time than a one-population GA search. Run time increases polynomially with the number of populations; equation of the regression curve is indicated for the strict and ring options only. (*b*) Run time (JC model, 160 taxa) vs. percent error for two-, three-, four-, six-, eight-, and 16-population metaGA searches under each of the five CP options. Coordinates of the one-population GA run are indicated. Probability and strict consensus options find excellent topologies as soon as the metaGA is run with more than three populations. The probability option is therefore optimal as it is significantly faster than the strict consensus option.

metaGA (i.e., CP with strict group consensus and four populations) reduces run time even further. The metaGA is up to seven times faster than StepAdd when using a simple model (JC) while it is over 800 times faster when considering a more realistic model of evolution (here, HKY + rate heterogeneity). Note that (*i*) much of the improvement seen under the complex models comes from taking an inter-step optimization strategy, (*ii*) the striking improvement in speed of the metaGA and GA over more classical search strategies is not at the expense of lower accuracy (see below), and (*iii*) the ratios of StepAdd to GA and StepAdd to metaGA run times increase with increasing number of taxa (JC, Fig. 2*b*). As the StepAdd algorithm yields a tree that is typically used as the starting point of a branch swapping search, we also show in Fig. 2*a* the times required for single rounds of NNI, SPR, and tree bisection-reconnection (TBR) branch swapping (9), i.e., the time required to swap to completion the single best ML tree. A typical heuristic run would comprise several rounds of SPR or TBR swapping on a starting tree obtained by StepAdd. Even if a reasonable starting tree could be obtained instantaneously, the minimum time required to TBR- or SPR-swap that tree would vastly exceed the time required by a full GA or metaGA search.

The efficiency of CP can be attributed to two factors. First, it allows the stopping rule to be reached more quickly (asterisks in Fig. 3*a*): near the end of the search, CP-constrained topological mutations affect a greatly reduced number of branches, hence swapping to completion is much faster than it would be on an unconstrained tree. Second, the consensus information shared among parallel populations allows them to increase their scores and the number of consensus partitions faster than if they were each searching in isolation (Fig. 3 *b* and *c*). Hence, despite the fact that a four-population metaGA search requires evaluation of four times more trees each generation than in a single GA search, the former completes the search much faster than the latter (Fig. 3*a*).

Accuracy of the MetaGA and Effect of the Number of Populations. Fig. 4 indicates that computing time increases with the number of populations involved in CP. As many as 8–12 populations are required to slow down the metaGA to computing times similar to those of single-population GA runs. However, as Fig. 4*b* indicates, two-population metaGA searches are not optimal as they yield trees with 2–4% error (in comparison with the ML score) for these idealized data sets. In this case, uninterrupted interaction between the same two populations does not allow CP to differentiate between informative and random consensus, such that the populations rapidly converge toward each other, terminating the search prematurely. This negative effect decreases dramatically with three populations and virtually disappears for runs using more than three populations (Fig. 4*b*): the likelihood that all populations evolve the same suboptimal partitions decreases very rapidly with the number of populations. In all analyses performed so far (data not shown), four-population METAPIGA runs seem to give the best compromise between computing speed and accuracy of topology inference.

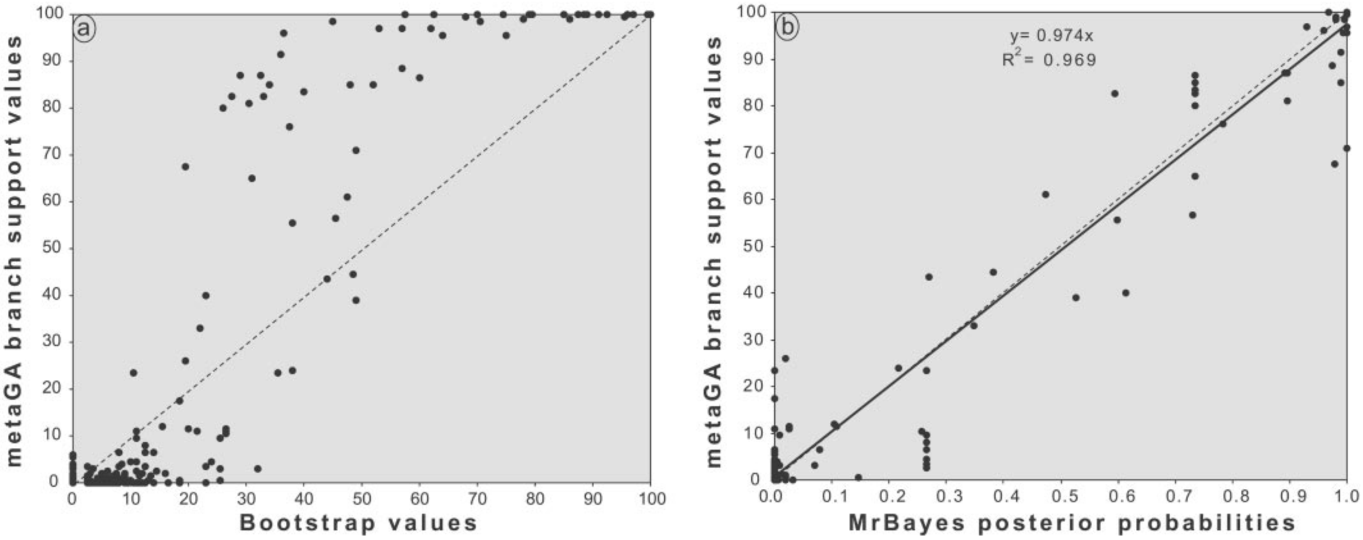

Fig. 5. MetaGA branch support values (10 runs with probability CP among 10 populations; each dot therefore indicates the number of trees, among the 100 best trees produced, exhibiting a given partition) vs. (*a*) bootstrap values (100 replicates; random starting tree, SPR branch swapping) and (*b*) MRBAYES posterior probabilities of partitions. A 55-taxa *rbc*L data set was analyzed under the JC model.

Fig. 4 indicates that the majority group consensus option of CP yields the worse scores without being particularly fast whereas the strict group consensus option is the most conservative, i.e., it yields the best scores but is also the slowest. The optimal combination of maximum speed and accuracy is probability consensus with four populations.

MetaGA Branch Support Values. Under CP, increasing the number of populations decreases the interdependence among the populations within a single run. Moreover, we would expect there to be a direct relationship between the proportion of populations that agree on a particular branch and the strength of support for that branch given the data set. Hence, it is conceivable that (*i*) a metaGA search with an infinite number of populations would produce the posterior probability distribution of possible trees, and (*ii*) a metaGA search with a finite number of populations would provide an estimate of this distribution. If both assumptions are correct, a set of multiple metaGA searches would produce trees and clades with frequencies that closely approximate their posterior probabilities, making the results of such a metaGA comparable to those provided by Bayesian approaches (13, 24). We tested this conclusion by comparing metaGA branch support values to bootstrap values as well as to posterior probabilities of clades produced by the program MRBAYES. Using an *rbc*L data set (15) comprised of 55 taxa and 1,314 bp, we performed 10 independent metaGA searches (strict CP among 10 populations) and computed the majority-rule consensus tree among the resulting 100 best trees (one tree per population). Remarkably, plotting metaGA support values against bootstrap proportions (100 replicates, with SPR branch swapping on a StepAdd starting tree) uncovers the same bias of bootstrap values (Fig. 5*a*) as that described with laboratory-generated phylogenies and simulations (26). This result suggests that every metaGA branch support value might efficiently approximate the probability of the corresponding clade being correct. Furthermore, Fig. 5*b* indicates that, under the specific conditions tested at least, there is a direct correspondence between frequencies of clades obtained under the metaGA and their posterior probabilities estimated with MRBAYES (n chains = 4, temperature = 0.05, sample frequency = 100, burn in = 25,000 steps, 5,000 samples taken). These preliminary results suggest that the frequencies with which trees and clades are sampled by using the metaGA might correspond to unbiased estimates of their posterior probabilities. Furthermore, given the results outlined in Fig. 4, the precision of the posterior probability estimator might be a function of metapopulation size.

We are grateful to Patrick Mardulyn and two anonymous reviewers for helpful comments on a previous version of this manuscript. This work was supported by the National Fund for Scientific Research Belgium (FNRS), the Free University of Brussels (ULB), the "Communauté Française de Belgique" (Grant ARC 98/03-223), and the Van Buuren, Brachet, and Defay Funds (Belgium).

1. Cook, S. (1971) in *Proceedings of the 3rd Annual Symposium on the Theory of Computing* (Association for Computing Machinery, New York), pp. 151–158.
2. Graham, R. L. & Foulds, L. R. (1982) *Math. Biosci.* **60,** 133–142.
3. Hillis, D. M. (1996) *Nature (London)* **383,** 130–131.
4. Huelsenbeck, J. P. (1995) *Syst. Biol.* **44,** 17–48.
5. Felsenstein, J. (1981) *J. Mol. Evol.* **17,** 368–376.
6. Steel, M. & Penny, D. (2000) *Mol. Biol. Evol.* **17,** 839–850.
7. Swofford, D. L. (2002) PAUP* (Sinauer, Sunderland, MA).
8. Felsenstein, J. (1993) PHYLIP (Univ. of Washington, Seattle).
9. Swofford, D. L., Olsen, G. J., Waddell, P. J. & Hillis, D. M. (1996) in *Molecular Systematics*, eds. Hillis, D. M., Moritz, C. & Mable, B. K. (Sinauer, Sunderland, MA), pp. 482–485.
10. Strimmer, K. & von Haeseler, A. (1996) *Mol. Biol. Evol.* **13,** 964–969.
11. Saitou, N. & Nei, M. (1987) *Mol. Biol. Evol.* **4,** 406–425.
12. Salter, L. A. & Pearl, D. K. (2001) *Syst. Biol.* **50,** 7–17.
13. Huelsenbeck, J. P., Ronquist, F. R., Nielsen, R. & Bollback, J. P. (2001) *Science* **294,** 2310–2314.
14. Matsuda, H. (1996) in *Pacific Symposium on Biocomputing '96*, eds. Hunter, L. & Klein, T. E. (World Scientific, London), pp. 512–523.
15. Lewis, P. O. (1998) *Mol. Biol. Evol.* **15,** 277–283.
16. Katoh, K., Kuma, K.-I. & Miyata, T. (2001) *J. Mol. Evol.* **53,** 477–484.
17. Foster, J. A. (2001) *Nat. Rev. Genet.* **2,** 428–436.
18. Hasegawa, M., Kishino, H. & Yano, T. (1985) *J. Mol. Evol.* **22,** 160–174.
19. Jukes, T. H. & Cantor, C. R. (1969) in *Mammalian Protein Metabolism*, ed. Munro, H. N. (Academic, New York), pp. 21–132.
20. Kimura, M. (1980) *J. Mol. Evol.* **16,** 111–120.
21. Van de Peer, Y., Neefs, J.-M., De Rijk, P. & De Wachter, R. (1993) *J. Mol. Evol.* **37,** 221–232.
22. Charleston, M. A. (2001) *J. Comput. Biol.* **8,** 79–91.
23. Felsenstein, J. & Churchill, G. A. (1996) *Mol. Biol. Evol.* **13,** 93–104.
24. Huelsenbeck, J. P. & Ronquist, F. R. (2002) *Biometrics*, in press.
25. Kendall, D. G. (1948) *Ann. Math. Stat.* **19,** 1–15.
26. Hillis, D. M. & Bull, J. J. (1993) *Syst. Biol.* **42,** 182–192.