(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
19 December 2013 (19.12.2013)

WIPO | PCT

(10) International Publication Number
WO 2013/188606 A2

(51) International Patent Classification:
A01H 1/04 (2006.01)

(21) International Application Number:
PCT/US2013/045538

(22) International Filing Date:
13 June 2013 (13.06.2013)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
61/660,055 15 June 2012 (15.06.2012) US

(71) Applicant: **AGRIGENETICS, INC.** [US/US]; 9330 Zionsville Road, Indianapolis, IN 46268-1054 (US).

(72) Inventors: **ROBBINS, Kelly, R.**; 5234 East 69th Street, Indianapolis, IN 46220 (US). **BACKLUND, Jan, Erik**; 370 East Arch Street, Indianapolis, IN 46202 (US).

(74) Agents: **EISENSCHENK, Frank, C.** et al.; Saliwanchik, Lloyd & Eisenschenk, P.O. Box 142950, Gainesville, FL 32614-2950 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))*

Published:

— *without international search report and to be republished upon receipt of that report (Rule 48.2(g))*

WO 2013/188606 A2

(54) Title: METHODS FOR SELECTION OF INTROGRESSION MARKER PANELS

(57) Abstract: This disclosure concerns marker-assisted plant selection and breeding. In specific embodiments, methods of identifying optimized marker panels for predicting the presence of a plant trait of interest, and/or marker panels thereby identified, are provided.

METHODS FOR SELECTION OF INTROGRESSION MARKER PANELS

FIELD OF THE DISCLOSURE

[0001] The present disclosure relates to plant breeding. More specifically, the disclosure relates to the use of an improved system for the identification and selection of a set of plant genetic markers that are highly useful for the introgression of a trait of interest.

BACKGROUND

[0002] The development of hybrid plant breeding has made possible considerable advances in quality and quantity of crops produced. Increased yield and the combination of desirable characteristics, such as resistance to disease and insects, heat and drought tolerance, and variations in plant composition are all possible, in part, due to plant hybridization procedures. Hybridization procedures rely on the contribution of pollen from a male parent plant to a female parent plant to produce the resulting hybrid.

[0003] The development of maize hybrids requires the development of homozygous inbred lines, the crossing of these lines, and the evaluation of the crosses. Pedigree breeding and recurrent selection are two breeding methods used to develop inbred lines from populations. Breeding programs combine desirable traits from two or more inbred lines, or various broad-based sources, into breeding pools from which new inbred lines are developed by selfing and selection of desired phenotypes. A hybrid maize variety is the cross of two such inbred lines, each of which may have one or more desirable characteristics absent in one of the inbred lines, or complementing characteristic(s) of the other. New inbred plants are crossed with other inbred lines, and the hybrids from these crosses are evaluated to determine which are desirable. Hybrid progeny of the first generation are designated F_1. The F_1 hybrid is typically more vigorous than its inbred parents. This hybrid vigor, termed heterosis, typically leads to, for example, increased vegetative growth and increased yield. Thus, in the development of hybrids, only the F_1 hybrids are generally sought.

[0004] To facilitate marker-assisted introgression, highly-informative markers (*e.g.*, SNP markers that are polymorphic between recurrent and donor parents) are desirable across populations targeted for trait introgression. The identification of a subset of informative markers can be viewed as a combinatorial problem. The solution is theoretically simple to achieve, requiring the evaluation of every possible combination, but computationally unfeasible. For example, current trait introgression applications using a marker set of 256 markers would require

a practitioner to evaluate 1.2 x 10^{279} marker combinations to exhaustively search all possible combinations, and this number increases exponentially as more markers are included in the marker set.

[0005] An ant colony algorithm (ACA) is a swarm intelligence algorithm that mimics the mechanism by which real ant colonies communicate in search of the best route to a food source. *See, e.g.*, Dorigio *et al.* (1999) Artificial Life 5(2):137-72. In nature, ants deposit chemical pheromones on the ground to form paths for other ants to follow. Initially, ants will disperse randomly from the nest in search of food, returning after a food source has been found. Ants finding the fastest route to a food source traverse the distance between the nest and the found food source at a faster rate, depositing more pheromone in the process. As the pheromone level builds, more ants preferentially choose the shorter path over longer paths with less pheromone, thereby depositing even more pheromone in the process. According to the foregoing, the natural biological behavior of ant colonies describes the fundamental elements of a positive feedback system, whereby all the ants of the colony will eventually select the best route from the nest to the food source.

[0006] Ant colony optimization (ACO) of solutions to problems with large sample spaces has been shown to be an efficient technique in communications network routing (Dorigio *et al.* (1999), *supra*; disease identification (Ressom *et al.* (2007) Bioinformatics 23(5):619-26); disease classification (Robbins *et al.* (2007) Math. Med. Biol. 24:413-26; and livestock genotyping (Spangler *et al.* (2009) Anim. Genet. 40:308-14).

BRIEF SUMMARY OF THE DISCLOSURE

[0007] Described herein are systems and methods for determining a set of genetic markers for use in plant breeding, the method comprising vectors, each storing one possible solution or "path". The method models communication between these vectors, called "ants", through an adaptive probability density function (PDF) referred to as a pheromone function. In embodiments, ants use the function to select subsets of genetic markers from the genome of a plant species of interest. The subsets of genetic markers may then be evaluated utilizing genetic information provided from a plurality of plants of interest. Based on the performance of the selected subset, the pheromone function may be updated, such that features yielding desirable solutions are more likely to be selected by ants in future iterations.

[0008] In particular embodiments, the ACA is adapted to identify a marker panel yielding optimal genome analysis coverage. In particular embodiments, the ACA is adapted to identify a marker panel yielding optimal linkage disequilibrium coverage.

[0009] In some embodiments, a TaqMan® SNP genotyping system (*e.g.*, an OpenArray® genotyping system) is used to provide genetic information from a plant of interest.

[0010] The foregoing and other features will become more apparent from the following detailed description of several embodiments, which proceeds with reference to the accompanying figures.

BRIEF DESCRIPTION OF THE FIGURES

[0011] **FIGs. 1a-y** include programming code that may be implemented on a computer to perform ant colony optimization according to particular embodiments.

[0012] **FIGs. 2-3** include comparisons of the performance of marker panels resulting from several optimization methods: for "ACA," sampling was based on the adaptive pheromone function; for "PS," sampling was based on the proportion of times a marker was informative (polymorphic between donor and recurrent parent) across all parent combinations, and for "RS," sampling was completely random. Performance (GA in **FIG. 1**; LD in **FIG. 2**) is expressed as the ratio of the coverage achieved by the selected marker panel and the coverage achieved when all markers are used (in this example, there were a total of 1371 markers). For each method, 24,000 marker subsets were evaluated, with only the top performing subset being presented.

[0013] **FIG. 2** includes plots of GA coverage for ACA, PS, and RS. GA coverage is presented as the proportion of coverage retained in the selected panel as compared to the coverage obtained when using all available markers.

[0014] **FIG. 3** includes plots of LD coverage for ACA, PS, and RS. LD coverage is presented as the proportion of coverage retained in the selected panel as compared to the coverage obtained when using all available markers.

[0015] **FIG. 4** includes plots of informative marker positions for the 256 markers selected by ACA (ACA Markers), and all available markers (All Markers). **FIG. 4a** includes a plot of marker positions for D020083/SLB01. **FIG. 4b** includes a plot of marker positions for SLD25BM/SLB01.

DETAILED DESCRIPTION

I. Overview of several embodiments

[0016] With the development of thousands of genetic markers (*e.g.*, SNP markers) for major crop species, panels of markers that will be effective across multiple crop lines for use in marker-assisted introgression of traits of interest (*e.g.*, agronomically important traits) now theoretically exist. Identification of such effective panels will allow for further automation and increased efficiency of the introgression process. However, given the prohibitively large and growing number of available markers associated with panel selection for plant breeding projects, exhaustive evaluation of all possible marker panels is computationally unfeasible. As such, systems and methods are provided herein to efficiently search an expansive sample space of markers to find an optimal solution in the form of an optimized marker panel. In some embodiments, systems and methods are provided that can identify informative marker subsets while searching only a small fraction of an immense sample marker space.

[0017] An ant colony optimization (ACO) system utilizes a positive feedback communication system that mimics the pheromone trails used by real ant colonies to find the best route to a food source, in order to efficiently search prohibitively large sample spaces for optimal solutions. In embodiments of the invention, an ACO system may be adapted to identify a panel of known and/or empirically determinable genetic markers that may yield optimized genome analysis (GA) and/or linkage drag (LD) coverage for use in marker-assisted plant breeding (*e.g.*, marker-assisted trait introgression). Using methods according to some embodiments of the invention, an ACO system is surprisingly effective at performing the task of identifying optimized marker panels for use in marker-assisted plant breeding, such that an ACO system consistently outperforms other optimization methods. The surprising superiority of methods according to particular embodiments may increase as the marker sample space increases.

[0018] In certain examples, an ACO was applied to identify a highly informative panel of 256 markers for plant breeding program development from a set of 1371 available SNP markers. In an application to 72 potential introgression projects, methods utilizing an ACO consistently outperformed all other tested methods. Using the identified set of 256 markers, 80% of the genome (GA) and LD coverage obtained when using all 1371 available markers was retained by the marker subset, further and quantitatively demonstrating the effectiveness of the methods utilizing an ACO.

II. Abbreviations

[0019]	ACA	ant colony algorithm
[0020]	ACO	ant colony optimization
[0021]	AFLP	amplified fragment length polymorphism
[0022]	DAF	DNA amplification fingerprinting
[0023]	GA	genome analysis
[0024]	LD	linkage drag
[0025]	PCR	polymerase chain reaction
[0026]	PDF	probability density function
[0027]	PS	sampling based on prior information
[0028]	QTL	quantitative trait locus
[0029]	RAPD	random amplification of polymorphic DNA
[0030]	RFLP	restriction fragment length polymorphism
[0031]	RS	random sampling
[0032]	SCAR	sequence characterized amplified region
[0033]	SNP	single nucleotide polymorphism

III. Terms

[0034] Ant: As used herein, an "ant" or "artificial ant" refers to an agent that moves from point to point. An "ant colony optimization" (ACO) system refers to a metaheuristic used for discrete, combinatorial optimization in some embodiments. In an ACO system, an ant may choose a next point to move to using a probabilistic function, both of trail accumulated on edges and of a heuristic value, which may be a function of the edge length. Ants will preferentially select discrete states with higher joint probabilities according to the pheromone function.

[0035] Backcrossing: Backcrossing methods may be used to introduce a nucleic acid sequence into plants. The backcrossing technique has been widely used for decades to introduce new traits into plants. Jensen, N., Ed. Plant Breeding Methodology, John Wiley & Sons, Inc., 1988. In a typical backcross protocol, the original variety of interest (recurrent parent) is crossed to a second variety (non-recurrent parent) that carries a gene of interest to be transferred. The resulting progeny from this cross are then crossed again to the recurrent parent, and the process is repeated until a plant is obtained wherein essentially all of the desired morphological and physiological characteristics of the recurrent plant are recovered in the converted plant, in addition to the transferred gene from the non-recurrent parent.

[0036] Genome analysis: "Genome analysis" generally refers to techniques that determine and compare genetic sequences. This includes DNA sequencing, routine use of DNA microarray technology for the analysis of gene expression profiles at the mRNA level and improved informatic tools to organize and analyze such data.

[0037] Isolated: An "isolated" biological component (such as a nucleic acid or protein) has been substantially separated, produced apart from, or purified away from other biological components in the cell of the organism in which the component naturally occurs (*i.e.*, other chromosomal and extra-chromosomal DNA and RNA, and proteins), while effecting a chemical or functional change in the component (*e.g.*, a nucleic acid may be isolated from a chromosome by breaking chemical bonds connecting the nucleic acid to the remaining DNA in the chromosome). Nucleic acid molecules and proteins that have been "isolated" include nucleic acid molecules and proteins purified by standard purification methods. The term also embraces nucleic acids and proteins prepared by recombinant expression in a host cell, as well as chemically-synthesized nucleic acid molecules, proteins, and peptides.

[0038] Linkage drag: "Linkage drag" refers to the length of donor genome segment surrounding a gene of introgression. The linkage drag segment is important as it may incorporate other less favourable alleles and drag them into the commercial population and the risk of this is related to its length. Molecular makers offer a tool in which the amount of wild or alien DNA can be monitored during each backcross generation.

[0039] Nucleic acid molecule: As used herein, the term "nucleic acid molecule" may refer to a polymeric form of nucleotides, which may include both sense and anti-sense strands of RNA, cDNA, genomic DNA, and synthetic forms and mixed polymers of the above. A nucleotide may refer to a ribonucleotide, deoxyribonucleotide, or a modified form of either type of nucleotide. A nucleic acid molecule as used herein is synonymous with "nucleic acid" and "polynucleotide." The term, nucleic acid molecule, includes single- and double-stranded forms of DNA. A nucleic acid molecule can include either or both of naturally occurring and modified nucleotides linked together by naturally occurring and non-naturally occurring nucleotide linkages. Nucleic acid molecules may be modified chemically or biochemically, or may contain non-natural or derivatized nucleotide bases, as will be readily appreciated by those of skill in the art. The term "nucleic acid molecule" also includes any topological conformation, including single-stranded, double-stranded, partially duplexed, triplexed, hairpinned, circular, and padlocked conformations.

[0040] Locus: As used herein, the term "locus" refers to a position on the genome that corresponds to a measurable characteristic (*e.g.*, a trait). An SNP locus is defined by a probe that hybridizes to DNA contained within the locus.

[0041] Marker: As used herein, a marker refers to a gene or nucleotide sequence that can be used to identify plants that are likely to have a particular allele and/or exhibit a particular trait or phenotype. A marker may be described as a variation at a given genomic locus. A genetic marker may be a short DNA sequence, such as a sequence surrounding a single base-pair change (single nucleotide polymorphism, or "SNP"), or a long sequence, for example, a minisatellite/simple sequence repeat ("SSR"). A "marker allele" refers to the version of the marker that is present in a particular plant. The term marker as used herein may refer to a cloned segment of plant chromosomal DNA, and may also or alternatively refer to a DNA molecule that is complementary to a cloned segment of plant chromosomal DNA.

[0042] In some embodiments, the presence of a marker in a plant may be detected through the use of a nucleic acid probe. A probe may be a DNA molecule or an RNA molecule. RNA probes can be synthesized by means known in the art, for example, using a DNA molecule template. A probe may contain all or a portion of the nucleotide sequence of the marker and additional, contiguous nucleotide sequence from the plant genome. This is referred to herein as a "contiguous probe." The additional, contiguous nucleotide sequence is referred to as "upstream" or "downstream" of the original marker, depending on whether the contiguous nucleotide sequence from the plant chromosome is on the 5' or the 3' side of the original marker, as conventionally understood. As is recognized by those of ordinary skill in the art, the process of obtaining additional, contiguous nucleotide sequence for inclusion in a marker may be repeated nearly indefinitely (limited only by the length of the chromosome), thereby identifying additional markers along the chromosome. Any and all of the above-described varieties of markers may be used in some embodiments of the present invention.

[0043] An oligonucleotide probe sequence may be prepared synthetically or by cloning. Suitable cloning vectors are well-known to those of skill in the art. An oligonucleotide probe may be labeled or unlabeled. A wide variety of techniques exist for labeling nucleic acid molecules, including, for example and without limitation: Radiolabeling by nick translation; random priming; tailing with terminal deoxytransferase; etc., where the nucleotides employed are labeled, for example, with radioactive ^{32}P. Other labels which may be used include, for example and without limitation: Fluorophores; enzymes; enzyme substrates; enzyme cofactors; enzyme inhibitors; etc.. Alternatively, the use of a label that provides a detectable signal, by

itself or in conjunction with other reactive agents, may be replaced by ligands to which receptors bind, where the receptors are labeled (for example, by the above-indicated labels) to provide detectable signals, either by themselves, or in conjunction with other reagents. *See*, *e.g.*, Leary *et al.* (1983) Proc. Natl. Acad. Sci. USA 80:4045-9.

[0044] A probe may contain a nucleotide sequence that is not contiguous to that of the original marker; this probe is referred to herein as a "noncontiguous probe." The sequence of the noncontiguous probe is located sufficiently close to the sequence of the original marker on the chromosome so that the noncontiguous probe is genetically linked to the same marker or gene as the original marker. For example, in some embodiments, a noncontiguous probe can be located within 500 kb; 450 kb; 400 kb; 350 kb; 300 kb; 250 kb; 200 kb; 150 kb; 125 kb; 120 kb; 100 kb; 0.9 kb; 0.8 kb; 0.7 kb; 0.6 kb; 0.5 kb; 0.4 kb; 0.3 kb; 0.2 kb; or 0.1 kb of the original marker on the chromosome.

[0045] A probe may be an exact copy of a marker to be detected. A probe may also be a nucleic acid molecule comprising, or consisting of, a nucleotide sequence that is substantially identical to a cloned segment of chromosomal DNA comprising a marker to be detected (for example, as defined by SNP ID in Table 2 (maize)).

[0046] A probe may also be a nucleic acid molecule that is "specifically hybridizable" or "specifically complementary" to an exact copy of the marker to be detected ("DNA target"). "Specifically hybridizable" and "specifically complementary" are terms that indicate a sufficient degree of complementarity such that stable and specific binding occurs between the nucleic acid molecule and the DNA target. A nucleic acid molecule need not be 100% complementary to its target sequence to be specifically hybridizable. A nucleic acid molecule is specifically hybridizable when there is a sufficient degree of complementarity to avoid non-specific binding of the nucleic acid to non-target sequences under conditions where specific binding is desired, for example, under stringent hybridization conditions.

[0047] Hybridization conditions resulting in particular degrees of stringency will vary depending upon the nature of the hybridization method of choice and the composition and length of the hybridizing nucleic acid sequences. Generally, the temperature of hybridization and the ionic strength (especially the Na^+ and/or Mg^{++} concentration) of the hybridization buffer will determine the stringency of hybridization, though wash times also influence stringency. Calculations regarding hybridization conditions required for attaining particular degrees of stringency are known to those of ordinary skill in the art, and are discussed, for example, in Sambrook *et al.* (ed.) Molecular Cloning: A Laboratory Manual, 2nd ed., vol. 1-3, Cold Spring

Harbor Laboratory Press, Cold Spring Harbor, NY, 1989, chapters 9 and 11; and Hames and Higgins (eds.) Nucleic Acid Hybridization, IRL Press, Oxford, 1985. Further detailed instruction and guidance with regard to the hybridization of nucleic acids may be found, for example, in Tijssen, "Overview of principles of hybridization and the strategy of nucleic acid probe assays," in Laboratory Techniques in Biochemistry and Molecular Biology- Hybridization with Nucleic Acid Probes, Part I, Chapter 2, Elsevier, NY, 1993; and Ausubel *et al.*, Eds., Current Protocols in Molecular Biology, Chapter 2, Greene Publishing and Wiley-Interscience, NY, 1995.

[0048] As used herein, "stringent conditions" encompass conditions under which hybridization will only occur if there is less than 25% mismatch between the hybridization molecule and the DNA target. "Stringent conditions" include further particular levels of stringency. Thus, as used herein, "moderate stringency" conditions are those under which molecules with more than 25% sequence mismatch will not hybridize; conditions of "medium stringency" are those under which molecules with more than 15% mismatch will not hybridize; and conditions of "high stringency" are those under which sequences with more than 10% mismatch will not hybridize. Conditions of "very high stringency" are those under which sequences with more than 6% mismatch will not hybridize.

[0049] In particular embodiments, stringent conditions are hybridization at 65 °C in 6x saline-sodium citrate (SSC) buffer, 5x Denhardt's solution, 0.5% SDS, and 100 μg sheared salmon testes DNA, followed by 15-30 minute sequential washes at 65 °C in 2x SSC buffer and 0.5% SDS, followed by 1x SSC buffer and 0.5% SDS, and finally 0.2x SSC buffer and 0.5% SDS.

[0050] With respect to all probes discussed, *supra*, the probe may comprise additional nucleic acid sequences, for example, promoters; transcription signals; and/or vector sequences.

[0051] As used herein, linkage between genes or markers refers to the phenomenon in which genes or markers on a chromosome show a measurable probability of being passed on together to individuals in the next generation. The closer two genes or markers are to each other, the closer to (1) this probability becomes. Thus, the term "linked" may refer to one or more genes or markers that are passed together with a second gene or marker with a probability greater than 0.5 (which is expected from independent assortment where markers/genes are located on different chromosomes). Because the proximity of two genes or markers on a chromosome is directly related to the probability that the genes or markers will be passed together to individuals in the next generation, the term "linked" may also refer herein to one or

more genes or markers that are located within about 2.0 Mb of one another on the same chromosome. Thus, two "linked" genes or markers may be separated by about 2.1 Mb; 2.00 Mb; about 1.95 Mb; about 1.90 Mb; about 1.85 Mb; about 1.80 Mb; about 1.75 Mb; about 1.70 Mb; about 1.65 Mb; about 1.60 Mb; about 1.55 Mb; about 1.50 Mb; about 1.45 Mb; about 1.40 Mb; about 1.35 Mb; about 1.30 Mb; about 1.25 Mb; about 1.20 Mb; about 1.15 Mb; about 1.10 Mb; about 1.05 Mb; about 1.00 Mb; about 0.95 Mb; about 0.90 Mb; about 0.85 Mb; about 0.80 Mb; about 0.75 Mb; about 0.70 Mb; about 0.65 Mb; about 0.60 Mb; about 0.55 Mb; about 0.50 Mb; about 0.45 Mb; about 0.40 Mb; about 0.35 Mb; about 0.30 Mb; about 0.25 Mb; about 0.20 Mb; about 0.15 Mb; about 0.10 Mb; about 0.05 Mb; about 0.025 Mb; about 0.012 Mb; and about 0.01 Mb. As used herein, the term "tightly linked" may refer to one or more genes or markers that are located within about 0.5 Mb of one another on the same maize chromosome. As used herein, the term "extremely tightly linked" may refer to one or more genes or markers that are located within about 100 kb of one another on the same maize chromosome.

[0052] As used herein, linkage between markers and traits or phenotypes of interest may refer to one or more markers that are each passed together with a trait or phenotype with a probability greater than expected from random chance (0.5). While a marker may be comprised in some examples within a gene that determines a particular trait or phenotype, it will be understood that most often a marker may be separated by a short distance (*e.g.*, less than about 2 Mb) from such a gene on the same chromosome. Moreover, it will be understood that most traits or phenotypes are polygenic, and thus a marker that is linked to a trait or phenotype may in some examples reside within, or be linked to, a QTL underlying a polygenic trait.

[0053] Linked, tightly linked, and extremely-tightly linked genetic markers may be useful in marker-assisted breeding programs, for example and without limitation, to introgress a trait or phenotype of interest into a plant variety; and to generate new plant varieties comprising a trait or phenotype of interest.

[0054] Marker-assisted breeding: As used herein, the term "marker-assisted breeding" may refer to an approach to breeding directly for one or more trait(s) (*e.g.*, a polygenic trait). In current practice, plant breeders attempt to identify easily detectable traits, such as flower color, seed coat appearance, or isozyme variants that are linked to an agronomically desired trait. The plant breeders then follow the agronomic trait in the segregating, breeding populations by following the segregation of the easily detectable trait. However, there are very few of these linkage relationships between traits of interest and easily detectable traits available for use in plant breeding. In some embodiments of the invention, marker-assisted breeding comprises

identifying one or more genetic markers (*e.g.*, SNP markers) that are linked to a trait of interest, and following the trait of interest in a segregating, breeding population by following the segregation of the one or more genetic markers. In some examples, the segregation of the one or more genetic markers may be determined utilizing a probe for the one or more genetic markers by assaying a genetic sample from a progeny plant for the presence of the one or more genetic markers.

[0055] Marker-assisted breeding provides a time- and cost-efficient process for improvement of plant varieties. Several examples of the application of marker-assisted breeding involve the use of isozyme markers. *See*, *e.g.*, Tanksley and Orton, eds. (1983) Isozymes in Plant Breeding and Genetics, Amsterdam: Elsevier. One example is an isozyme marker associated with a gene for resistance to a nematode pest in tomato. The resistance, controlled by a gene designated *Mi*, is located on chromosome 6 of tomato and is very tightly linked to Aps1, an acid phosphatase isozyme. Use of the *Aps1* isozyme marker to indirectly select for the *Mi* gene provided the advantages that segregation in a population can be determined unequivocally with standard electrophoretic techniques; the isozyme marker can be scored in seedling tissue, obviating the need to maintain plants to maturity; and co-dominance of the isozyme marker alleles allows discrimination between homozygotes and heterozygotes. *See* Rick (1983) in Tanksley and Orton, *supra*.

[0056] Optimized: As used herein in the context of a panel of genetic markers, the term "optimized" refers to a panel of markers that performs better (*e.g.*, provides greater GA or LD coverage) than a reference panel comprising the same number of non-identical markers in predicting the presence or absence of a trait of interest. Thus, in some examples, an "optimized" marker panel is a subset of a large number of genetic markers in a plant species that performs better in predicting the presence or absence of a trait of interest or donor DNA than a different subset of the same size consisting of markers from the large number of genetic markers. In some examples, an "optimized" marker panel is a subset of a set of genetic markers in a plant species that retains more of the predictive value (for the presence or absence of a trait of interest) of the entire set of genetic markers than a different subset of the same size consisting of markers from the entire set of genetic markers.

[0057] The term "optimized" may refer to a subset that provides the best performance over all other subsets, but this is not necessarily the case. An optimized marker set may be further optimized to provide even better performance, for example, by performing further

iterations of an ACO system, or by performing iterations of an ACO system in the presence of additional segregation data.

[0058] Sequence identity: The term "sequence identity" or "identity," as used herein in the context of two nucleic acid sequences, may refer to the nucleobases in the two sequences that are the same when aligned for maximum correspondence over a specified comparison window.

[0059] As used herein, the term "percentage of sequence identity" may refer to the value determined by comparing two optimally aligned nucleic acid sequences over a comparison window, wherein the portion of the sequence in the comparison window may comprise additions or deletions (*i.e.*, gaps) as compared to the reference sequence (which does not comprise additions or deletions) for optimal alignment of the two sequences. The percentage is calculated by determining the number of positions at which the identical nucleobase occurs in both sequences to yield the number of matched positions, dividing the number of matched positions by the total number of positions in the comparison window, and multiplying the result by 100 to yield the percentage of sequence identity.

[0060] Methods for aligning sequences for comparison are well-known in the art. Various programs and alignment algorithms are described in, for example: Smith and Waterman (1981) Adv. Appl. Math. 2:482; Needleman and Wunsch (1970) J. Mol. Biol. 48:443; Pearson and Lipman (1988) Proc. Natl. Acad. Sci. U.S.A. 85:2444; Higgins and Sharp (1988) Gene 73:237-44; Higgins and Sharp (1989) CABIOS 5:151-3; Corpet *et al.* (1988) Nucleic Acids Res. 16:10881-90; Huang *et al.* (1992) Comp. Appl. Biosci. 8:155-65; Pearson *et al.* (1994) Methods Mol. Biol. 24:307-31; Tatiana *et al.* (1999) FEMS Microbiol. Lett. 174:247-50. A detailed consideration of sequence alignment methods and homology calculations can be found in, *e.g.*, Altschul *et al.* (1990) J. Mol. Biol. 215:403-10.

[0061] The National Center for Biotechnology Information (NCBI) Basic Local Alignment Search Tool (BLAST™; Altschul *et al.* (1990)) is available from several sources, including the National Center for Biotechnology Information (Bethesda, MD), and on the internet, for use in connection with several sequence analysis programs. A description of how to determine sequence identity using this program is available on the internet under the "help" section for BLAST™. For comparisons of nucleic acid sequences, the "Blast 2 sequences" function of the BLAST™ (Blastn) program may be employed using default parameters. Nucleic acid sequences with even greater similarity to the reference sequences will show increasing percentage identity when assessed by this method.

[0062] As used herein, the term "substantially identical" may refer to nucleotide sequences that are more than 85% identical. For example, a substantially identical nucleotide sequence may be at least 85.5%; at least 86%; at least 87%; at least 88%; at least 89%; at least 90%; at least 91%; at least 92%; at least 93%; at least 94%; at least 95%; at least 96%; at least 97%; at least 98%; at least 99%; or at least 99.5% identical to the reference sequence.

[0063] Single-nucleotide polymorphism (SNP): As used herein, the term "single-nucleotide polymorphism" may refer to a DNA sequence variation occurring when a single nucleotide in the genome (or other shared sequence) differs between members of a species or paired chromosomes in an individual.

[0064] Within a population, SNPs can be assigned a minor allele frequency of the lowest allele frequency at a locus that is observed in a particular population. This is simply the lesser of the two allele frequencies for single-nucleotide polymorphisms. There are variations between plant populations, so an SNP allele that is common in one population may be rarer in a different population.

[0065] Single nucleotide polymorphisms may fall within coding sequences of genes, non-coding regions of genes, or in the intergenic regions between genes. SNPs within a coding sequence will not necessarily change the amino acid sequence of the protein that is produced, due to degeneracy of the genetic code. An SNP in which both forms lead to the same polypeptide sequence is termed "synonymous" (sometimes called a silent mutation). If a different polypeptide sequence is produced, they are termed "non-synonymous." A non-synonymous change may either be missense or nonsense, where a missense change results in a different amino acid, and a nonsense change results in a premature stop codon. SNPs that are not in protein-coding regions may still have consequences for gene splicing, transcription factor binding, or the sequence of non-coding RNA. SNPs are usually biallelic and thus easily assayed in plants and animals. Sachidanandam (2001) Nature 409:928-33.

[0066] Stigmergy: As used herein, the term "stigmergy" or "stigmergic communication" refers to indirect communication between agents mediated by physical modifications of environmental state variables, the values of which are only locally accessible by the communicating agents (*i.e.*, ants).

[0067] Trait or phenotype: The terms "trait" and "phenotype" are used interchangeably herein. For the purposes of the present disclosure, traits of particular interest include agronomically important traits, as may be expressed, for example, in a crop plant.

IV. Markers for use in plant breeding

[0068] Embodiments of the invention include genetic markers in a plant that may be linked to a trait of interest. Some embodiments include a set of markers in the genome of a plant from which may be identified, through the implementation of an ACO system, a subset of markers that may be used to predict the presence or absence of a trait of interest in a plant from which a genetic sample has been provided. Sets of genetic markers, and optimized subsets identified therefrom, may comprise one or more markers that are individually linked to the trait of interest.

[0069] Some markers that may be used in particular embodiments are known in the art. For example, genetic markers have been made available in many plant species through genome sequencing, genotyping, and QTL mapping studies. Additional markers that may be used in particular embodiments may be identified by any technique known to those of skill in the art, including for example and without limitation: molecular techniques such as RAPD, identification of RFLPs, AFLP-PCR, DAF, identification of SCARs, and/or identification of microsatellites; and direct comparison of aligned genomic nucleic acid sequences from several populations.

[0070] In some examples, a set of markers comprises SNP markers. The genotyping of a plant for one or more SNP markers is easily carried out, for example, by using one of many PCR-based analysis techniques. In particular examples, the genotyping of a plant over a set of SNP markers may be carried out by utilizing the OpenArray® SNP genotyping system (Applied BioSystems). The OpenArray® system uses "chips" comprising a panel of SNPs to determine the genotype of an organism from which a genetic sample has been provided, by assaying for specific hybridization of nucleic acids within the genetic sample to the panel of SNPs.

V. Ant colony optimization

[0071] ACO is an optimization method that was designed by reference to the natural process of pheromone use in ants to identify the shortest path from a spatial point of interest to the nest. Dorigo and Gambardella (1997) BioSystems 43:73-81; Dorigo *et al.* (1999) Artificial Life 5:137-72. In nature, ants each deposit a certain amount of pheromone while walking, and each ant probabilistically prefers to follow a direction rich in pheromone. If an obstacle appears along a path between a spatial point of interest (*e.g.*, a food source) and the nest, those ants approaching the obstacle must choose between turning right or left to avoid the obstacle. In the absence of a pheromone cue providing direction one way or the other, half the ants will choose

to turn right, and the other half will choose to turn left. Those ants which choose, by chance, the shorter path around the obstacle will reconstitute an uninterrupted pheromone trail more rapidly than those who choose a longer path. This behavior establishes an autocatalytic process, whereby the shorter path receives a greater amount of pheromone per time unit, and a larger number of ants consequently chooses the shorter path. If this process is allowed to reach its natural conclusion, all the ants will rapidly choose the shorter path.

[0072] The shortest path around the obstacle may be thought of as an emergent property of the interaction between the shape of the obstacle and the distributed behavior of the ants. Although all the ants move at approximately the same speed and deposit a pheromone trail at approximately the same rate, it takes longer to contour obstacles on their longer side than on their shorter side. Consequently, the pheromone trail accumulates more rapidly on the shorter side. The ants' preference for a trail with higher pheromone levels makes this accumulation still quicker on the shorter path. According to the foregoing, while each ant may find a solution (*i.e.*, a path between the two points), only the activity of a collection of $\boldsymbol{m}$ ants leads to optimization.

[0073] In an exemplary ACO system, $\boldsymbol{m}$ ants are placed on randomly selected spatial points comprised within an appropriate representation of the problem to be solved. At each time step, the ants move to new points and may modify the pheromone trail on the edges (*i.e.*, paths between points) used, in a process referred to as "local trail updating." When all the ants have completed a movement, the ant that made the shortest movement may modify the edge belonging to its movement ("global trail updating") by adding an amount of pheromone trail that is inversely proportional to the movement length. In some embodiments, ants may be able to determine the distance between points, and/or have a working memory ($\boldsymbol{M_k}$) used to memorize points already visited (the working memory may be emptied at the beginning of each new movement, and may be updated after each time step).

[0074] Ants are tasked to find a shortest path joining an initial problem situation to a destination situation. Ants must move step-by-step through adjacent problem states. Ants build solutions through applying a probabilistic decision policy to move through adjacent states, in most embodiments making use only of local information without prediction of future states. Thus, the decision policy may be completely local in space and time. The decision policy is a function of the *a priori* information represented by the problem specifications, and local modifications in the problem environment (pheromone trails) induced by ants in the past. Once an individual ant has built a problem solution and deposited pheromone information, the ant may be deleted from the ACO system. Although the complexity of each ant is such that it can build a

feasible solution (as a real ant can somehow find a path between the nest and the food), high-quality solutions are the result of the cooperation among the individuals of the whole colony.

[0075] In some embodiments, features of an ACO system of the invention may include, for example and without limitation, a plurality of cooperating individual agents (a colony of ants); an artificial "pheromone" trail (*i.e.*, numeric information that takes into account the depositing ant's current history or performance and can be read/written by any ant accessing the state) that modifies the local state of a problem for local stigmergetic communication; a sequence of local moves to find shortest paths; and a stochastic decision policy using local information without prediction of future states. In some embodiments, features of an ACO system of the invention may also include, for example and without limitation, a discrete problem environment comprising discrete adjacent states, wherein the ants' movements consist of transitions between discrete adjacent states; internal states in each ant comprising memory of the ant's past actions; and deposition of pheromone in an amount that is a function of the quality of the solution found.

[0076] In some embodiments, stigmergetic communication provided by local pheromone trails may be the only communication channel between the ants. However in some embodiments, some prediction of future states may be employed. Michel and Middendorf (1998) "An island model based Ant System with lookahead for the shortest supersequence problem." In Proceedings of PPSN-V, Fifth International Conference on Parallel Problem Solving from Nature, Eiben *et al.* (Eds.), Springer-Verlag, Berlin.

[0077] In some embodiments, a stochastic component of the ants' decision policy and/or an "evaporation mechanism" may prevent ants from being constrained by past decisions to rapidly migrate toward a same previously-visited part of the search space. An evaporation mechanism modifies the information in local pheromone trails over time, so that the ant colony may forget or partially forget its past history. A stochastic component of the ants' decision policy determines the balance between the exploration of new points in the state space and the exploitation of accumulated knowledge according to the level of stochasticity in the policy, and the strength of the updates in the local pheromone trails. A particular level of stochasticity and/or a strength of pheromone trail updates, as well as the strength of an evaporation mechanism, may be determined in embodiments according to the discretion of the practitioner.

[0078] In some examples, the ants' timing in deposition of pheromone is problem dependent. For example, ants may update pheromone trails only after having generated a solution. Also in some examples, an ACO system may be enriched with capabilities including,

for example and without limitation, local optimization (*see*, *e.g.*, Dorigo and Gambardella (1997) IEEE Transactions on Evolutionary Computation 1(1):53-66); backtracking/recovery procedures (*see* Di Caro and Dorigo (1998) J. Art. Intel. Res. (JAIR) 9:317-65); and an extra-ant component that may observe the ants' behavior to collect useful global information and deposit additional pheromone information that biases the ants' search processes from a non-local perspective (Dorigo *et al.* (1999), *supra*). These and other modifications may improve, for example, the efficiency and/or performance of the ACO system. In an ACO system, ant generation and activity, pheromone evaporation, and extra-ant activity may be synchronized during the performance of the system. In some examples, a sequential scheduling of system activities is used.

[0079] In embodiments of the invention, an optimized panel of genetic markers from a plant is identified through the implementation of an ACO system. In some embodiments, the spatial points in the problem space may correspond to discrete subsets of markers from a larger discrete marker set. In some embodiments, the stigmergetic communication between the ants may be represented by a PDF that is updated by pheromone levels that are determined by the genome (GA) and linkage drag (LD) coverage provided by the selected markers for a trait of interest. Performance of an ACO system according to such embodiments through multiple time steps may identify a panel of genetic markers that is optimized for the identification and/or introgression of the trait of interest. In particular examples, the larger discrete marker set may comprise at least about 500 markers; at least about 600 markers; at least about 700 markers; at least about 800 markers; at least about 900 markers; at least about 1000 markers; at least about 1100 markers; at least about 1200 markers; at least about 1300 markers; at least about 1400 markers; at least about 1500 markers; at least about 1600 markers; at least about 1700 markers; at least about 1800 markers; at least about 1900 markers; at least about 2000 markers; or more.

[0080] To test the effectiveness of an ACO system in identifying an optimized panel of genetic markers from a plant, the ACO may be applied to multiple populations of the plant that are targeted for introgression of the trait of interest. In particular examples, the ACO may be applied to more than about 100 populations; less than about 100 populations; less than about 90 populations; less than about 80 populations; less than about 75 populations; less than about 70 populations; less than about 60 populations; less than about 50 populations; or fewer. The effectiveness of the ACO system may be evaluated by comparing the GA and LD coverage obtained using an identified, optimized marker subset, to the coverage obtained when using all of the markers in the larger set from which the optimized marker subset was identified. Thus

evaluated, the effectiveness of the ACO system may be compared to alternative panel selection methods. Such an optimized marker subset may provide better GA and/or LD coverage than alternative methods across a number of trait introgression projects.

[0081] General information regarding ACO systems and their implementation may be found in, for example, Dorigo *et al.* (1999), *supra*.

VI. Use of optimized marker panels

[0082] Some embodiments include methods of identifying plants likely to comprise a trait of interest using optimized molecular marker panels that have been identified by a process utilizing an ACO system. In particular embodiments, nucleic acid molecules (*e.g.*, genomic DNA or mRNA) may be extracted from a plant. The extracted nucleic acid molecules may then be contacted with one or more probes that are specifically hybridizable to markers in an optimized marker panel. Specific hybridization of the one or more probes to the extracted nucleic acid molecules is indicative of the presence of the trait of interest in the plant. Such methods may result in a cost savings for plant developers, because use of such methods may eliminate the need to phenotype individual plants generated during development (for example, by crossing a plant variety having a trait of interest with a plant variety lacking the trait of interest).

[0083] In some embodiments, optimized marker panels that have been identified by a process utilizing an ACO system to be predictive of a trait of interest may be used to transfer segment(s) of DNA that contain one or more genes or QTLs that determine or contribute to the trait of interest (*i.e.*, trait introgression). In particular embodiments, a method for using such an optimized marker panel may comprise, for example and without limitation, providing a first parent plant comprising markers in the optimized marker panel; providing a second parent plant; analyzing the genomic DNA of the first and second parent plants with probes that are specifically hybridizable to markers in the optimized marker panel; crossing the two parental plant genotypes to obtain a progeny population, and analyzing those progeny for the presence of the markers in the optimized marker panel; backcrossing the progeny that contain the markers in the optimized marker panel to the second parental genotype to produce a first backcross population, and then continuing with a backcrossing program until a final progeny is obtained that comprises any desired trait(s) exhibited by the second parent genotype and the markers in the optimized marker panel, thereby transferring the segment(s) of DNA that contain one or more genes or QTLs that determine or contribute to the trait of interest. In particular

embodiments, the progeny of the first cross, or any subsequent backcross, may be crossed to a third plant that is of a different line or genotype than either the first or second plant. A final progeny plant that comprises any desired trait(s) exhibited by the second parent genotype and the markers in the optimized marker panel may be likely to comprise the trait of interest.

[0084] In some examples, individual progeny obtained in each crossing and backcrossing step are selected by marker panel analysis at each generation. In some examples, analysis of the genomic DNA of the two parent plants with probes that are specifically hybridizable to the markers in the optimized marker panel reveals that one of the parent plants comprises fewer of the markers to which the probes specifically hybridize, or none of the markers to which the probes specifically hybridize. In some examples, the first parent plant may comprise the trait of interest, or may lack the trait of interest but comprise a genotype that is predictive of the trait of interest.

[0085] According to the foregoing, progeny plants may be subjected in some examples to a genotype and/or zygosity determination. Once progeny plants have been genotyped, and/or their zygosity has been determined, the skilled artisan may select those progeny plants that have a desired genetic composition (*e.g.*, progeny plants that comprise markers of an optimized marker panel). Such selected progeny plants may be used in further crosses, selfing, or cultivation. Methods of trait introgression that employ optimized marker panels identified by a process utilizing an ACO system to be predictive of the trait may reduce or eliminate the cultivation and/or reproduction of plants that do not have the desired genetic composition, and thereby provide desirable reliability and predictability (through expected Mendelian patterns of inheritance) in selective plant breeding or development programs.

[0086] The following examples are provided to illustrate certain particular features and/or embodiments. The examples should not be construed to limit the disclosure to the particular features or embodiments exemplified.

EXAMPLES

Example 1: Materials and Methods

[0087] Data. The dataset consisted of genotype information on 72 recurrent inbred maize lines targeted for trait introgression, and five inbred maize lines serving as donors. Each line was genotyped for 1371 markers available for use with the OpenArray® SNP genotyping

system. For each recurrent and donor parent combination, markers were scored as either informative or uninformative based on polymorphisms between the two parents.

[0088] SNP panel selection. Three sampling methods were used to select subsets of markers ($\boldsymbol{S_k}$): random sampling (RS), sampling based on prior information (PS) (prior information is computed as the polymorphic rate of a SNP), and the ACA. RS sampled marker subsets at random, and the probability of a marker being selected by PS was based on the proportion of times a marker was informative for the 72 recurrent and donor parent combinations. The ACO sampling method used an ant colony sampling method. Each set of markers was evaluated based on the GA and LD coverage, calculated as:

$$\text{coverage}_{\text{GA}} = \frac{\sum_{i=1}^{nm} \text{MWGA}_i}{\text{genome length}} \quad (1)$$

and

$$\text{coverage}_{\text{LD}} = \frac{\sum_{i=1}^{nmi} \text{MWLD}_i}{\sum_{j=1}^{ni} \text{length insert chromosome}_j} \quad (2)$$

where ***nm*** is the number of informative markers in $\boldsymbol{S_k}$; ***nmi*** is the number of informative markers flanking an insert site; ***ni*** is the number of chromosomes with trait inserts; and MWGA_i and MWLD_i are marker weights for GA and LD, respectively, subject to the following restrictions:

$$\text{MWGA}_i = \begin{cases} 20\text{cM} & \text{if weight of marker}_i > 20\text{cM} \\ \text{marker weight}_i & \text{otherwise} \end{cases}$$

For MWLD_i, if marker_i was further than 30 cM from the insert site,

$$\text{MWLD}_i = \begin{cases} 10\text{cM} & \text{if weight of marker}_i > 10\text{cM} \\ \text{marker weight}_i & \text{otherwise} \end{cases},$$

otherwise

$$\text{MWLD}_i = \begin{cases} 5\text{cM} & \text{if weight of marker}_i > 5\text{cM} \\ \text{marker weight}_i & \text{otherwise} \end{cases}$$

[0089] Marker weights were calculated as the sum of half the distance (in cM) between the marker of interest and the nearest informative upstream and downstream markers in $\boldsymbol{S_k}$.

[0090] Ant Colony Optimization. Artificial ants were defined as parallel units that communicate through a probability density function (PDF) that is updated by weights or "pheromone levels," which in this case are determined by the GA and LD coverage provided by the selected markers. *See* Dorigo *et al.* (1999), *supra*; *see also* Ressom *et al.* (2007), *supra*; *see*

also Robbins *et al.* (2007) Math. Med. Biol. 24:413-26. The probability of sampling marker m at time t was defined as:

$$P_m(t) = \frac{(\tau_m(t))^{\alpha}\eta_m^{\beta}}{\sum_{m=1}^{nf}(\tau_m(t))^{\alpha}\eta_m^{\beta}} \quad (3)$$

where $\tau_m(t)$ is the amount of pheromone for marker m (out of a total of nf markers) at time t, η_m is the prior information used by PS for marker m; α and β are parameters determining the weight given to pheromone deposited by ants and *a priori* information on the features, respectively. For this study, α and β were both set to 1.

[0091] The ACA was initialized with all markers having an equal baseline level of pheromone used to compute used to compute $P_m(0)$ for all markers. Using the PDF as defined in Eq. 3, each of j ants will select a subset (S_k) of n markers from the sample space S containing all 1371 markers. The pheromone level of each marker m in S_k was then updated according to the performance of S_k as:

$$\tau_m(t+1) = (1-\rho) * \tau_m(t) + \Delta\tau_m(t) \quad (4)$$

where ρ is a constant between 0 and 1 that represents the rate at which the pheromone trail evaporates; $\Delta\tau_m(t)$ is the change in pheromone level for marker *m* based on the performance of S_k, and it is set to 0 if feature $m \notin S_k$. This process is repeated for all S_k.

[0092] The procedure employed can be summarized by the following steps: First, each ant selects a predetermined number of markers; then, using the selected markers, performance is calculated as:

$$\text{performance} = 0.5 * \text{coverage}_{\text{GA}} + 0.5 * \text{coverage}_{\text{LD}} \quad (5)$$

and third, the change in pheromone is calculated as:

$$\Delta\tau_m(t) = \text{performance}^{(1-\text{performance})} \quad (6)$$

[0093] Following the update of pheromone levels according to Eq. 4, the PDF is updated according to Eq. 3, and the process is repeated until a convergence criteria is met, which in this example was a predefined number of iterations. As the PDF is updated, the selected features that perform better are sampled at higher likelihoods by subsequent artificial ants, which, in turn, deposit more "pheromone," thus leading to a autocatalytic positive feedback system. **FIG. 1** sets forth programming code used in this example to perform the above-described ant colony optimization procedure.

Example 2: Improved Performance of Ant Colony Optimization Systems

[0094] GA and LD coverage for marker panels selected using ACO, PS, and RS were determined. **FIGs. 2-3**. The ACO outperformed both PS and RS for all marker panel sizes tested, which was a clear indication that the adaptive sampling method of the ACO yields superior panel selections. At 256 markers, the panel selected by ACO for the GA and LD traits recovered 80% of the coverage obtained when using all available markers. Achieving this level of coverage using only 19% of the markers (256/1371 markers) is quite surprising and remarkable, particularly in view of the immense size of the sample space. Furthermore, the ACO converged to stable solutions in less than 5 minutes, indicating that larger sample spaces could be accommodated by the system.

Example 3: Optimized SNP Panel

[0095] The ACO was employed to select a 256 SNP panel for use in the TaqMan® OpenArray® SNP genotyping system. Given the importance of good LD coverage and the cost of large gaps in GA coverage, the criteria used for testing ACO performance were modified to place more weight on LD coverage and a higher penalty for large gaps in GA coverage. The new evaluation criteria calculated LD coverage using only the markers 25 cM upstream and downstream of inserts. For GA coverage, weights of markers covering more than 40 cM were set to 0, rather than 20 cM as previously described.

[0096] Using the new criteria, the ACA recovered 75% and 87% of the GA and LD coverage obtained using all available markers, respectively. Plots of the positions of informative markers can be found in **FIG. 4** for two selected populations. While some large gaps in coverage are present, it can be seen that the gaps in the ACA panel correspond to gaps present when all informative markers are used. In general, the ACA panel yielded remarkably even coverage relative to the coverage obtained when using all available markers.

CLAIMS

What may be claimed is:

1. A method of determining a set of biological markers for identification of a plant likely to comprise a trait of interest, the method comprising:

utilizing an ant colony optimization system to identify an optimized subset of plurality of genetic markers that is predictive of the trait of interest, wherein the optimized subset of the plurality of genetic markers is the set of biological markers for identification of a plant likely to comprise the trait of interest.

2. The method according to claim 1, further comprising:

providing an inbred plant that does not comprise the trait of interest, wherein the inbred plant comprises a genotype for the plurality of genetic markers;

crossing-a first donor plant comprising a genotype for the plurality of genetic markers with the inbred plant to produce a progeny plant, wherein the progeny plant comprises a genotype for the plurality of genetic markers, and determining whether the progeny plant comprises the trait of interest; and

providing a database comprising a plurality of genotypes for the plurality of genetic markers, wherein each genotype is the genotype of a progeny plant produced by crossing the inbred plant with an additional different donor plant.

3. The method according to claim 2, further comprising:

providing a genetic sample from the first donor plant; and

genotyping the first donor plant for the plurality of genetic markers.

4. The method according to claim 1, wherein utilizing an ant colony optimization system comprises:

defining a problem space comprised of discrete adjacent subsets of the plurality of genetic markers, and a plurality of agents, wherein the agents choose in successive time steps between the discrete adjacent subsets according to a probability density function that is updated

over the successive time steps by a value determined by the genome (GA) and linkage drag (LD) coverage for the trait of interest provided by the chosen discrete adjacent subsets; and

allowing the agents to choose discrete adjacent subsets of the plurality of genetic markers over a predetermined number of successive time steps.

5. The method according to claim 4, wherein the discrete adjacent subset chosen in the last of the predetermined number of successive time steps is the set of biological markers for identification of a plant likely to comprise the trait of interest.

6. The method according to claim 1, wherein the genotype of the inbred plant for the plurality of genetic markers is determined by genotyping.

7. The method according to claim 1, wherein the genotype of the progeny plant for the plurality of genetic markers is determined by genotyping.

8. The method according to claim 1, wherein the genotype of an additional different donor plant is determined by genotyping.

9. The method according to claim 1, wherein the plurality of genetic markers comprise SNP markers.

10. The method according to claim 9, wherein the plurality of genetic markers consist of SNP markers.

11. The method according to claim 1, wherein the plant is selected from a group comprising maize, soybean, tobacco, carrot, canola, rapeseed, cotton, palm, peanut, *Oryza sp.*, *Arabidopsis sp.*, *Ricinus sp.*, and sugarcane.

12. The method according to claim 1, wherein the plurality of genetic markers comprises at least about 1000 markers.

13. A set of markers determined by the method according to claim 1.

14. The set of markers of claim 13, wherein the set comprises less than about 300 markers.

15. A method of identifying a plant likely to comprise a trait of interest, the method comprising:

providing the set of markers of claim 13;

providing a genetic sample comprising nucleic acids from a plant; and

contacting the nucleic acids with probes that are specifically hybridizable to markers in the set of markers, wherein specific hybridization of the probes to the nucleic acids is indicative of the presence of the trait of interest in the plant.

16. The method according to claim 15, wherein the markers comprise SNP markers.

17. The method according to claim 15, wherein the plant is selected from a group comprising maize, soybean, tobacco, carrot, canola, rapeseed, cotton, palm, peanut, *Oryza sp.*, *Arabidopsis sp.*, *Ricinus sp.*, and sugarcane.

18. A method of transferring a plant trait of interest, the method comprising:

providing the set of markers of claim 13;

providing a first parent plant comprising the trait of interest;

providing a second parent plant lacking the trait of interest;

analyzing the genomic DNA of the first and second parent plants with probes that are specifically hybridizable to markers in the set of markers, thereby determining the genotype of the first and second parent plants for the markers in the set of markers;

crossing the two parental plant genotypes to obtain a progeny population;

analyzing plants of the progeny population with probes that specifically hybridize to markers in the set of markers, thereby determining the genotype of the progeny plants for the markers in the set of markers;

backcrossing a progeny plant that comprises the same genotype as the first parent plant for the markers in the set of markers to the second parental genotype to produce a first backcross population; and

continuing with a backcrossing program until a final progeny plant is obtained that comprises any desired trait(s) exhibited by the second parent genotype and the same genotype

as the first parent plant for the markers in the set of markers, thereby transferring the trait of interest.

19. The method according to claim 18, wherein the progeny of the first cross, or any subsequent backcross in the backcrossing program, is crossed to a third parent plant comprising a different genotype than either the first parent plant or second parent plant.

20. The method according to claim 18, wherein individual progeny obtained in each crossing and backcrossing step are genotyped for the markers in the set of markers.

21. The method according to claim 18, wherein the markers comprise SNP markers.

22. The method according to claim 18, wherein the plants are selected from a group comprising maize, soybean, tobacco, carrot, canola, rapeseed, cotton, palm, peanut, *Oryza sp.*, *Arabidopsis sp.*, *Ricinus sp.*, and sugarcane.

FIG. 1a.

```
#include <iostream>
#include <iomanip>
#include <stdio.h>
#include <fstream>
#include <sstream>
#include <cmath>
#include <time.h>
#include <gsl/gsl_randist.h>
#include <gsl/gsl_cdf.h>
#include "basicfun.h"

using namespace std;
//double gsl_cdf_qaussian_P(double x, double sigma);
int iAntCount;
int iLociCount;
int iIndv;
int iLociModel;
int iItCount;
double alpha;
double beta;
double rou;
double phe;
int method;
int iTopModel;
int iTopLoci;
int nInserts;
double* eva_TopModel;
int** loci_TopModel;
using namespace std;
int rnd(int uper)
{
      return (rand()%uper);
}
double rnd(int low, double uper)
{
      double p=(rand()/(double)RAND_MAX)*((uper)-(low))+(low);
      return (p);
}
double* calcVarMean(double* a){
    int i;
    double* b;
    b=new double[2];
    double sumA=0.0;
    double sumAS=0.0;
    for(i=0;i<iIndv;i++){
        sumA=sumA+a[i];
        sumAS=sumAS+(a[i]*a[i]);
    }
    b[0]=sumA/iIndv;
    b[1]=(sumAS+(sumA/iIndv))/(iIndv-1);
```

FIG. 1b.

```
    return(b);

}
int* sortVector(int* a)
{
        int* b;
        b=new int[iLociModel];

        for(int i=0;i<iLociModel;i++)
        {
                b[i]=a[i];
        }
        int ind=1;
        for(int i=0;i<iLociModel;i++)
        {
                ind=1;
                for(int j=0;j<i;j++)
                {
                        if(a[i]<b[j])
                        {
                                int tracker=i;
                                while(tracker>j){
                                        b[tracker]=b[tracker-1];
                                        tracker=tracker-1;
                                }
                                b[j]=a[i];
                                ind=-1;
                                break;
                        }
                }
                if(ind==1){
                        b[i]=a[i];
                }

        }

        return(b);
}
int char2int(char a)
{
      int b;
      b=int(a)-48;
      return b;
}
double char2double(char a)
{
      double b;
      b=double(int(a)-48);
      return b;
}
```

FIG. 1c.

```
class SNPdata
{
      public:
            double avgEva;
        double* phenotype;
      int** SNP;
      double *pheromone;
      int* chromosome;
      int* insertChromosome;
        double* insertPosition;
        double* position;
        double* chrLen;
        double genomeLen;
        double insertChrLen;
        void inputdata(char *path);
      void initiate();
      void deldata();
      void display_pheromone();
      void setpheromone();
      double* prior;
};
void SNPdata::setpheromone()
{
      int i;
        for(i=0;i<iLociCount;i++)
      pheromone[i]=phe;

}
void SNPdata::inputdata(char *path)
{
        avgEva=0.0;
        cout<<"Reading data from "<<path<<endl;
        ifstream in(path);
        int i,j;
        char temp[5000];
        char*
pathp="/home/u399178/ACA/library/lib/ACApackage/prior.txt";
        char*
pathm="/home/u399178/ACA/library/lib/ACApackage/SNPMap.txt";
        char*
pathc="/home/u399178/ACA/library/lib/ACApackage/chrLengths.txt";
        string line;
        string linep;
        string linem;
        string linem2;
        string linem3;
        string linem4;
        string linec;
        getline(in,line);
        istringstream values(line);
        string word;
        string wordp;
```

FIG. 1d.

```
string wordm;
string wordm2;
string wordm3;
string wordm4;
string wordc;
nInserts=3;
//parsing phenotype values
i=0;
while(!values.eof())
{
        getline(values,word,',');
        i++;
}
iIndv=i;
cout<<"Number of samples:"<<iIndv<<endl;
//parsing genotypes
j=0;
while(!in.eof())
{
        in.getline(temp,5000);
        j++;
}
iLociCount=j-1;
pheromone=new double[iLociCount];
prior=new double[iLociCount];
//chromosome=new int[iLociCount];
//position=new double[iLociCount];
//insertChromosome=new int[nInserts];
//insertPosition=new double[nInserts];
//chrLen=new double[20];

cout<<"Number of loci:"<<iLociCount<<endl;
SNP=new int*[iLociCount];
for(i=0;i<iLociCount;i++)
{
        SNP[i]=new int[iIndv];
        prior[i]=1.0;
}
phenotype=new double[iIndv];
in.close();
ifstream in1(path);
getline(in1,line);
istringstream values1(line);
float f;
cout<<"populating vector with phenotypic values"<<endl;
//populating vector with phenotypic values
i=0;
while(!values1.eof())
{
        getline(values1,word,',');
```

FIG. 1e.

```
                istringstream float_iss(word);
                float_iss>>f;
                phenotype[i]=f;
                i++;
        }
        //populating vector with prior information and
        // SNP map information
        cout<<"populating vector with prior information"<<endl;
        ifstream inp(pathp);
        getline(inp,linep);
        istringstream valuesp(linep);

        ifstream inm(pathm);
        getline(inm,linem);
        getline(inm,linem2);
        getline(inm,linem3);
        getline(inm,linem4);

        istringstream valuesm(linem);
        istringstream valuesm2(linem2);
        istringstream valuesm3(linem3);
        istringstream valuesm4(linem4);

        float fp;
        float fm;
        i=0;
        /*while(!valuesp.eof())
        {
                //getline(valuesp,wordp,',');
                //istringstream float_iss(wordp);
                //float_iss>>fp;
                prior[i]=1.0;

                if(method==4){
                    getline(valuesm,wordm,',');
                    istringstream float_ism(wordm);
                    float_ism>>fm;
                    chromosome[i]=int(fm);

                    getline(valuesm2,wordm2,',');
                    istringstream float_ism2(wordm2);
                    float_ism2>>fm;
                    position[i]=fm;
                }
//              cout<<"chr "<<chromosome[i]<<" position
"<<position[i]<<endl;
                i++;
        }
        //reading in insert information
        cout<<"reading insert information"<<endl;
```

FIG. 1f.

```
        if(method==4){
            i=0;
             while(!valuesm3.eof())
             {
                getline(valuesm3,wordm3,',');
                istringstream float_ism(wordm3);
                float_ism>>fm;
                insertChromosome[i]=int(fm);

                getline(valuesm4,wordm4,',');
                istringstream float_ism2(wordm4);
                float_ism2>>fm;
                insertPosition[i]=fm;
//                cout<<"chr "<<insertChromosome[i]<<" position
"<<insertPosition[i]<<endl;
                i++;
            }

             // reading in chromosome lengths
            cout<<"reading chromosome lengths"<<endl;
            ifstream inc(pathc);
            getline(inc,linec);
            istringstream valuesc(linec);
            i=0;
            genomeLen=0.0;
            insertChrLen=0.0;
            float fc;
            while(!valuesc.eof())
            {
                getline(valuesc,wordc,',');
                istringstream float_iss(wordc);
                float_iss>>fc;
                chrLen[i]=fc;
                genomeLen=genomeLen+chrLen[i];
                for(j=0;j<nInserts;j++){
                    if(insertChromosome[j]==i+1){
                        insertChrLen=insertChrLen+chrLen[i];
                    }
                }
                i++;
            }
        }*/
        //populating matrix with genotypic values
        cout<<"populating matrix with genotypic values"<<endl;
        j=0;
        //while(!in1.eof())
        while(!in1.eof())
        {
                in1.getline(temp,5000);
                if(j!=iLociCount)
```

FIG. 1g.

```
                {

                        for(i=0;i<iIndv;i++)
                        {

                            SNP[j][i]=char2int(temp[i]);

                        }
                        j++;
                }
        }


}
void SNPdata::display_pheromone()
{
      int i;
      for(i=0;i<iLociCount;i++)
      {
            std::cout<<setprecision(3)<<pheromone[i];
            cout<<" ";
      }
      cout<<endl;
}
SNPdata testdata;
class ant
{
      private:
      double* prob;
      int m_iLociCount;
      int* AllowedLoci;
      public:
      int ant_number;
      void addlocus(int locus);
      int* tabu;
      void display();
      void initiate();
      void start();
      void move();
      int ChooseNextLocus();
      void destroy();
};
void ant::initiate()
{
      int i;
      tabu=new int[iLociModel+1];

      prob=new double[iLociCount];
      AllowedLoci=new int[iLociCount];
}
```

FIG. 1h.

```
void ant::start()
{
	m_iLociCount=0;
	int i;
	for(i=0;i<iLociCount;i++)
	{
		AllowedLoci[i]=1;
		prob[i]=0;
	}
}
int ant::ChooseNextLocus()
{
	int i,j;
	j=-1;
	double temp=0;
	int curLocus=tabu[m_iLociCount-1];
	for(i=0;i<iLociCount;i++)
	{
		prob[i]=0;
		if(AllowedLoci[i]==1)
		{

	prob[i]=pow(testdata.pheromone[i],alpha);//*pow(testdata.prior[i]
,beta);
			//prob[i]=pow(testdata.prior[i],1.0);
				//cout<<i<<' '<<prob[i]<<endl;
				temp+=prob[i];
		}
	}
	double mRate,mSelect;
	while(j==-1)
	{
		mRate=rnd(0,temp);
		mSelect=0;
		for(i=0;i<iLociCount;i++)
		{
			if(AllowedLoci[i]==1)
			mSelect+=prob[i] ;
			if(mSelect>=mRate)
			{
				j=i;
				break;
			}
		}
	}
return j;
}
void ant::addlocus(int locus)
{
	tabu[m_iLociCount]=locus;
```

FIG. 1i.

```
      m_iLociCount++;
      AllowedLoci[locus]=0;
}
void ant::move()
{
      int j;
      j=ChooseNextLocus();
      addlocus(j);
}
void ant::destroy()
{
      delete[] tabu;
      delete[] prob;
      delete[] AllowedLoci;
}
double* LDCoverage(int* a)
{
        int lower;
        int upper;
        int indUpper;
        int indLower;
        double positionLower;
        double positionUpper;
        double coverage;
        double* b;
        double insertCoverageLength;
        b=new double[iIndv];


        for(int j=0;j<iIndv;j++)
        {
                insertCoverageLength=0.0;
                b[j]=0.0;
                for(int i=0;i<iLociModel;i++)
                {
                    for(int k=0;k<nInserts;k++){

if(testdata.chromosome[a[i]]==testdata.insertChromosome[k]){
                            //cout<<testdata.chromosome[a[i]]<<'
'<<testdata.insertChromosome[k]<<endl;
                            //cout<<testdata.insertPosition[k]<<endl;

                            lower=i-1;
                            upper=i+1;
                            indLower=1;
                            indUpper=1;

                            if(testdata.SNP[a[i]][j]==1){
                                 coverage=0.0;
                                 while(indLower<=1)
```

FIG. 1j.

```
                                {

                                        if(lower<0){
                                                positionLower=0;
                                                indLower=2;
                                                break;
                                        } else
if(testdata.chromosome[a[lower]]!=testdata.chromosome[a[i]]){
                                                positionLower=0;
                                                indLower=2;
                                                break;
                                        } else {

if(testdata.SNP[a[lower]][j]==1){

positionLower=testdata.position[a[lower]];
                                                        indLower=2;
                                                        break;
                                                } else {
                                                        lower=lower-1;
                                                }
                                        }
                                }

                                while(indUpper<=1)
                                {

                                        if(upper>=iLociModel){

positionUpper=testdata.chrLen[testdata.chromosome[a[i]]-1];
                                            indUpper=2;
                                            break;
                                        } else
if(testdata.chromosome[a[upper]]!=testdata.chromosome[a[i]]){

positionUpper=testdata.chrLen[testdata.chromosome[a[i]]-1];
                                                indUpper=2;
                                                break;
                                        } else {

if(testdata.SNP[a[upper]][j]==1){

positionUpper=testdata.position[a[upper]];
                                                        indUpper=2;
                                                        break;
                                                } else {
                                                        upper=upper+1;
                                                }
                                        }
                                }
```

FIG. 1k.

```
                                    coverage=.5*(testdata.position[a[i]]-
positionLower)+.5*(positionUpper-testdata.position[a[i]]);
                                    //if(testdata.position[a[i]]-
testdata.insertPosition[k]>-25.0&&testdata.position[a[i]]-
testdata.insertPosition[k]<25.0){

insertCoverageLength=insertCoverageLength+coverage;
                                        if(coverage>10.0){
                                            coverage=10.0;
                                        }
                                    //} else {
                                      //  coverage=0.0;

                                    //}
                                    b[j]=b[j]+coverage;
                                }
                            }
                        }
                    }
                    b[j]=b[j]/testdata.insertChrLen;
                    //if(insertCoverageLength==0.0){
                      //  b[j]=0.001;
                    //} else {
                      //
if(insertCoverageLength<(nInserts*60))insertCoverageLength=nInserts*60;
                        //b[j]=b[j]/insertCoverageLength;
                    //}
                    // cout<<j<<" "<<b[j]<<endl;
        }
        return(b);
}
double* genomeCoverage(int* a)
{
        int lower;
        int upper;
        int indUpper;
        int indLower;
        double positionLower;
        double positionUpper;
        double coverage;
        double* b;
        b=new double[iIndv];


        for(int j=0;j<iIndv;j++)
        {
                b[j]=0.0;
                for(int i=0;i<iLociModel;i++)
                {
```

FIG. 11.

```
                        lower=i-1;
                        upper=i+1;
                        indLower=1;
                        indUpper=1;

                        if(testdata.SNP[a[i]][j]==1){
                                coverage=0.0;
                                while(indLower<=1)
                                {

                                        if(lower<0){
                                                positionLower=0;
                                                indLower=2;
                                                break;
                                        } else
if(testdata.chromosome[a[lower]]!=testdata.chromosome[a[i]]){
                                                positionLower=0;
                                                indLower=2;
                                                break;
                                        } else {

if(testdata.SNP[a[lower]][j]==1){

positionLower=testdata.position[a[lower]];
                                                        indLower=2;
                                                        break;
                                                } else {
                                                        lower=lower-1;
                                                }
                                        }
                                }

                                while(indUpper<=1)
                                {

                                    if(upper>=iLociModel){

positionUpper=testdata.chrLen[testdata.chromosome[a[i]]-1];
                                                indUpper=2;
                                                break;

                                        } else
if(testdata.chromosome[a[upper]]!=testdata.chromosome[a[i]]){

positionUpper=testdata.chrLen[testdata.chromosome[a[i]]-1];
                                                indUpper=2;
                                                break;

                                        } else {
```

FIG. 1m.

```
if(testdata.SNP[a[upper]][j]==1){

positionUpper=testdata.position[a[upper]];
                                                        indUpper=2;
                                                        break;
                                                } else {
                                                        upper=upper+1;
                                                }
                                        }

                                        }

                                        coverage=.5*(testdata.position[a[i]]-
positionLower)+.5*(positionUpper-testdata.position[a[i]]);
                                    // if(coverage>40){
                                    //     coverage=0.0;
                                    // }
                                     if(coverage>20.0){
                                            coverage=20.0;
                                     }
                                     b[j]=b[j]+coverage;
                            }
                    }
                    //cout<<b[j]<<" "<<testdata.genomeLen<<endl;
                    b[j]=b[j]/testdata.genomeLen;

        }
        return(b);
}
double impurityDetection(int* a)
{
    int comp=0;
    int track=0;
    double rand=0.0;
    double* trackIndv=new double[iIndv];
    double startTrackIndv = 0.0;
    double* detections=new double[iIndv];
    double sumDetections=0.0;
    double sumComparisons = 0.0;
    double detectionRate = 0.0;
    for(int j=0;j<iIndv;j++){
        trackIndv[j]=startTrackIndv;
        startTrackIndv+=1.0;
    }

    for(int j=0;j<iIndv;j++){

        track=0;
       // cout<<"j "<<j<<endl;
        //randomly selecting another line to tes for polymorphisms with
```

FIG. 1n.

```
the current line
        while(track==0){
            rand=rnd(0,(iIndv-1));
          //  cout<<rand<<endl;
            for(int k=0;k<iIndv;k++){

              if(k==(iIndv-1)){
                  comp=k;
              } else if(rand>=trackIndv[k]&&rand<trackIndv[(k+1)]){
                  comp=k;
                  k=iIndv;
              }

            }

            if(comp!=j){
                track=1;
            }

        }
        //cycling through selected snp to identify polymorphisms
        for(int i=0;i<iLociModel;i++){


if(testdata.SNP[a[i]][j]!=0&&testdata.SNP[a[i]][comp]!=0){
                  if(testdata.SNP[a[i]][j]!=testdata.SNP[a[i]][comp]){
                      detections[j]=1.0;
                      i=iLociModel;
                  }

              }
        }
    }
    //calculating the proportion of times an impurity is detected
    for(int j=0;j<iIndv;j++){
        sumDetections=sumDetections+detections[j];
        sumComparisons=sumComparisons+1;
    }
    detectionRate=sumDetections/sumComparisons;
    return(detectionRate);
}
double evaluation(int* loci)
{


        double eva;
        double pval;
        int i,j,k,l,m;
        double** locidata;
        double** y;
        double** reddata;
```

FIG. 1o.

```
        int* keep;
        double* GACov;
        double* LDCov;
        int val;
        int numcol;
        int numrows;
        int* freq;
        int* sortedLoci;
        double* mvGA;
        double* mvLD;
        mvGA=new double[2];
        mvLD=new double[2];
        double zLD = 0.0;
        double zGA = 0.0;
        double pValGA= 0.0;
        double pValLD= 0.0;

        sortedLoci=sortVector(loci);


        if(method==4){
            double mGALD=0.0;
            double vGALD=0.0;
            double score=0.0;

                GACov=genomeCoverage(sortedLoci);
                LDCov=LDCoverage(sortedLoci);
                mvGA=calcVarMean(GACov);
                mvLD=calcVarMean(LDCov);

                //zGA=(mvGA[0]-.5)/pow((mvGA[1]/iIndv),.5);
                //zLD=(mvLD[0]-.5)/pow((mvLD[1]/iIndv),.5);
                //pValGA=gsl_cdf_gaussian_P(zGA,1.0);
                //pValLD=gsl_cdf_gaussian_P(zLD,1.0);
                mGALD=.5*mvGA[0]+.5*mvLD[0];
                vGALD=.5*mvGA[1]+.5*mvLD[1];
                eva=mGALD;
                //eva=0.0*(mGALD/(mGALD+vGALD))+1.0*mGALD;

                //testdata.avgEva=testdata.avgEva+(eva/24000.0);
                //cout<<mvGA[0]<<" "<<mvGA[1]<<" "<<mvLD[0]<<"
"<<mvLD[1]<<endl;
                //cout<<eva<<endl;
                //for(i=0; i<iIndv; i++){
                //    cout<<GACov[i]<<" "<<LDCov[i]<<endl;
                //}

        } else if(method==5){
            //cout<<"starting evaluation"<<endl;
            eva=impurityDetection(sortedLoci);
```

FIG. 1p.

```
            eva=eva*eva*eva;
            //cout<<"detection rate "<<eva<<endl;
        } else {

            l=(int)pow(3.0,iLociModel);

            keep=new int[iIndv];
            locidata=new double*[iIndv];
            numrows=iIndv;

            freq=new int[l];
           for(i=0;i<l;i++)
           {
                 freq[i]=0;
           }
           for(i=0;i<iIndv;i++)
           {
                 locidata[i]=new double[l];

                 for(j=0;j<l;j++)
                 {
                       locidata[i][j]=0;
                 }
                 val=0;
                        for(j=0;j<iLociModel;j++){
                       val+=(int)((testdata.SNP[loci[j]][i]-
1)*pow(3.0,iLociModel-1-j));
                        }
                        keep[i]=val;
                        locidata[i][val]=1;
                 freq[val]++;
           }
           numcol=0;
           for(i=0;i<l;i++)
           {
                    if(freq[i]>1){
                 numcol++;
                    } else {
                        for(j=0;j<iIndv;j++){
                            if(keep[j]==i){
                                keep[j]=-1;
                                numrows=numrows-1;
                            }
                        }
                    }

           }

                y=new double*[numrows];
                m=0;
```

FIG. 1q.

```
            for(i=0;i<iIndv;i++)
        {
                if(keep[i]!=-1){
                    y[m]=new double[1];
                    y[m][0]=testdata.phenotype[i];
                    m=m+1;
                }
            }

            reddata=new double* [numrows];
        for(i=0;i<numrows;i++)
        {
             reddata[i]=new double[numcol];

        }

            k=0;
        for(i=0;i<l;i++)
        {
             if(freq[i]>1)
             {
                        m=0;
                        for(j=0;j<iIndv;j++)
                    {
                            if(keep[j]!=-1){
                                reddata[m][k]=locidata[j][i];

                                m=m+1;
                            }

                        }
             k++;
             }
        }

    }

    if(method==1)
    {

        if(numrows>1&&numcol>1){

            eva=linear_regression(reddata,y,numrows,numcol);
          //  cout<<eva<<" "<<numrows<<" "<<numcol<<endl;
        } else {
            eva=0;
        }

    }
    if(method==2)
```

FIG. 1r.

```
{
    if(numrows>1&&numcol>1){
        eva=logistic_regression(reddata,y,numrows,numcol);
    } else {
        eva=0;
    }

}
if(method==3)
{

    if(numrows>1&&numcol>1){
        //cout<<"1a"<<endl;
        eva=crossvalidation(reddata,y,numrows,numcol);
        //cout<<"2a"<<endl;
    } else {
        eva=0;
    }

}

int tag=0;
if(eva>eva_TopModel[0])
{
        tag=1;
        for(i=1;i<iTopModel;i++)
        {
                if(eva==eva_TopModel[i])
                {
                        tag=0;
                        break;
                }
        }
}

if(tag==1)
{
        eva_TopModel[0]=eva_TopModel[1];
        for(j=0;j<iLociModel;j++)
        loci_TopModel[0][j]=loci_TopModel[1][j];
        i=1;
        while(i<iTopModel && eva>eva_TopModel[i])
        {
                eva_TopModel[i-1]=eva_TopModel[i];
                for(j=0;j<iLociModel;j++)
                loci_TopModel[i-1][j]=loci_TopModel[i][j];
                i++;
        }

        eva_TopModel[i-1]=eva;
```

FIG. 1s.

```
                for(j=0;j<iLociModel;j++){
                    loci_TopModel[i-1][j]=sortedLoci[j];
                }

        }
       // cout<<"1"<<endl;
 for(i=0;i<numrows;i++)
 {
    delete [] y[i];
    delete [] reddata[i];
 }

 delete [] y;
 delete [] reddata;
 for(i=0;i<iIndv;i++){
     delete [] locidata[i];
 }

 delete [] locidata;
 delete [] mvGA;
 delete [] mvLD;
 delete [] sortedLoci;
 delete [] keep;
 delete [] freq;
// cout<<"2"<<endl;
}
class project
{
      public:
       void UpdatePheromone();
       void GetAnt();
       void StartSearch();
       ant* ants;
       project();
       ~project();
};
void project::UpdatePheromone()
{
        double meanAccuracy;
        double countAnts=0.0;
        int i,j,k;
      for(i=0;i<iLociCount;i++)
            testdata.pheromone[i]=testdata.pheromone[i]*(1-rou);
      double detaphe, accurate;
      int* locidata;
      locidata=new int[iLociModel];
      for(i=0;i<iAntCount;i++)
      {
            for(j=1;j<=iLociModel;j++)
            {
```

FIG. 1t.

```
                locidata[j-1]=ants[i].tabu[j];
            }
///call function to calculate accuracy///
            accurate=evaluation(locidata);
                meanAccuracy=meanAccuracy+accurate;
                countAnts=countAnts+1.0;
                detaphe=pow(accurate,(1-accurate));
            for(j=1;j<=iLociModel;j++)
            {
                 testdata.pheromone[ants[i].tabu[j]]+=detaphe;
            }
      }
meanAccuracy=meanAccuracy/countAnts;
cout<<meanAccuracy<<endl;
}
project::project()
{
      ants=new ant[iAntCount];
      int i;
      for(i=0;i<iAntCount;i++)
            {ants[i].initiate();}
}
project::~project()
{

      delete [] ants;
}
void project::GetAnt()
{
       int i=0;
       int locus;
      srand((unsigned)time(NULL)+rand());
      for (i=0;i<iAntCount;i++)
      {
//          {

              ants[i].start();
              locus=rnd(iLociCount);
              ants[i].ant_number=i;
              ants[i].addlocus(locus);
//           }

      }
}
void project::StartSearch()
{
      int max,i,j;
      max=0;
      double temp;
      while(max<iItCount)
```

FIG. 1u.

```
	{
		cout<<"Iteration:"<<max<<endl;
		GetAnt();
		for(i=0;i<iAntCount;i++)
		{
			for(j=1;j<=iLociModel;j++)
			{
				ants[i].move();
				if(i==1){
					//cout<<"locus "<<j<<"
"<<ants[i].tabu[j]<<endl;
				}
			}

		}

		UpdatePheromone();
		max++;
	}
}
char inputfile[300];
char outputfile[300];
void loadparameters(char* path);
void writeresult();
int main()
{
	int i,j,k;
	iAntCount=1000;
	iItCount=500;
	alpha=1;
	beta=.5;
	rou=0.05;
	phe=1;
	iLociModel=2;
	method=1;
	iTopModel=100;
	iTopLoci=100;
	char*
parapath="/home/u399178/ACA/library/lib/ACApackage/parameters.txt";
	loadparameters(parapath);
	cout<<"parameters:"<<endl;
	cout<<"Number of ants:"<<iAntCount<<endl;
	cout<<"Number of iterations:"<<iItCount<<endl;
	cout<<"Alpha:"<<alpha<<endl;
	cout<<"Rou:"<<rou<<endl;
	cout<<"Initial pheromone level:"<<phe<<endl;
	cout<<"Locus model:"<<iLociModel<<endl;
	cout<<"Method:"<<method<<endl;
	cout<<"Number of top interactions:"<<iTopModel<<endl;
	eva_TopModel=new double[iTopModel];
```

FIG. 1v.

```
      loci_TopModel=new int*[iTopModel];
      for(i=0;i<iTopModel;i++)
      {
            loci_TopModel[i]=new int[iLociModel];
            eva_TopModel[i]=0;
      }
      testdata.inputdata(inputfile);
      testdata.setpheromone();
      project pjt;
      pjt.StartSearch();
      writeresult();
      return 0;
}
void writeresult()
{
      ofstream out(outputfile);
      int i,j,temp1;
      double temp2;
      out<<"Top interactions:"<<endl;
      out<<"Evaluation\tloci\n";
      //change to write out all loci in top model
        for(i=iTopModel-1;i>=0;i--)
      {
            out<<eva_TopModel[i];
                        for(j=0;j<(iLociModel-1);j++){
                            out<<" "<<loci_TopModel[i][j];
                        }
                  out<<" "<<loci_TopModel[i][iLociModel-1]<<endl;
      }
      int tag[iLociCount];
      double SortPheromone[iLociCount];
      for(i=0;i<iLociCount;i++)
      {
            tag[i]=i;
            SortPheromone[i]=testdata.pheromone[i];
      }
      out<<"Top ranking pheromones"<<endl;
      out<<"Locus\tpheromone"<<endl;
      for(i=0;i<iTopLoci;i++)
      {
            for(j=i+1;j<iLociCount;j++)
            {
                  if(SortPheromone[j]>SortPheromone[i])
                  {
                        temp1=tag[i];
                        tag[i]=tag[j];
                        tag[j]=temp1;
                        temp2=SortPheromone[i];
                        SortPheromone[i]=SortPheromone[j];
                        SortPheromone[j]=temp2;
```

FIG. 1w.

```
                }
            }
            out<<tag[i]<<" "<<SortPheromone[i]<<endl;
      }
      out<<"Locus\tPheromone"<<endl;
      for(i=0;i<iLociCount;i++)
      {
            out<<i<<"\t"<<testdata.pheromone[i]<<endl;
      }
}
void loadparameters(char* path)
{
//cout<<"hello"<<endl;
        FILE *f = fopen(path, "r");
        if(f == NULL)
        {       printf("Cannot open the parameter file \"%s\"\n",
path);
                return;
        }
        int i;
        char tmp[1000];
        while(fgets(tmp, 1000, f) != NULL)
        {
                char str[1000];
                sprintf(str, "%s", tmp);
                str[10]=0;
                if(strcmp(str, "iLociModel") == 0)
                {
                        for(i = 10; i < (int)strlen(tmp); i++)
                                if(tmp[i] >= 48 && tmp[i] < 58)
                                        break;
                        iLociModel=atoi(&tmp[i]);
                }
                str[9] = 0;
                if(strcmp(str, "iAntCount") == 0)
                {       for(i = 9; i < (int)strlen(tmp); i++)
                                if(tmp[i] >= 48 && tmp[i] < 58)
                                        break;
                        iAntCount=atoi(&tmp[i]);
                }
                if(strcmp(str, "iTopModel") == 0)
                {       for(i = 9; i < (int)strlen(tmp); i++)
                                if(tmp[i] >= 48 && tmp[i] < 58)
                                        break;
                        iTopModel=atoi(&tmp[i]);
                }
                str[8]=0;
                if(strcmp(str, "iItCount") == 0)
                {       for(i = 8; i < (int)strlen(tmp); i++)
                                if(tmp[i] >= 48 && tmp[i] < 58)
                                        break;
```

FIG. 1x.

```
                        iItCount=atoi(&tmp[i]);
                }
                if(strcmp(str, "iTopLoci") == 0)
                {       for(i = 8; i < (int)strlen(tmp); i++)
                                if(tmp[i] >= 48 && tmp[i] < 58)
                                        break;
                        iTopLoci=atoi(&tmp[i]);
                }
                str[7] = 0;
                if(strcmp(str, "INPFILE") == 0)
                {
                for(i = 7; i < (int)strlen(tmp); i++)
                    if(tmp[i] != ' ' && tmp[i] != '\t')
                            break;
                    int j;
                    for(j = i + 1; j < (int)strlen(tmp); j++)
                         if(tmp[j] == ' ' || tmp[j] == '\t' || tmp[j]
== '"' || tmp[j]=='\n')
                         break;
                    tmp[j] = 0;
                    if(tmp[i] == '"') sprintf(inputfile, "%s", &tmp[i +
1]);
                        else sprintf(inputfile, "%s", &tmp[i]);
                }
                else if(strcmp(str, "OUTFILE") == 0)
                {
                for(i = 7; i < (int)strlen(tmp); i++)
                         if(tmp[i] != ' ' && tmp[i] != '\t')
                         break;
                    int j;
                for(j = i + 1; j < (int)strlen(tmp); j++)
                    if(tmp[j] == ' ' || tmp[j] == '\t' || tmp[j] == '"'
|| tmp[j]=='\n')
                    break;
                 tmp[j] = 0;
                if(tmp[i] == '"') sprintf(outputfile, "%s", &tmp[i +
1]);
                        else sprintf(outputfile, "%s", &tmp[i]);
                }
                str[6]=0;
                if(strcmp(str, "method") == 0)
                {
                        for(i = 6; i < (int)strlen(tmp); i++)
                                if(tmp[i] >= 48 && tmp[i] < 58)
                                        break;
                        method=atoi(&tmp[i]);
                }
                str[5] = 0;
                if(strcmp(str, "alpha") == 0)
                {
                        for(i = 5; i < (int)strlen(tmp); i++)
```

FIG. 1y.

```
                        if(tmp[i] >= 48 && tmp[i] < 58)
                                break;
                alpha=atof(&tmp[i]);
        }
        str[3]=0;
        if(strcmp(str, "rou") == 0)
        {       for(i = 3; i < (int)strlen(tmp); i++)
                        if(tmp[i] >= 48 && tmp[i] < 58)
                                break;
                rou = atof(&tmp[i]);
        }
        else if(strcmp(str, "phe") == 0)
        {       for(i = 3; i < (int)strlen(tmp); i++)
                        if(tmp[i] >= 48 && tmp[i] < 58)
                                break;
                phe = atof(&tmp[i]);
        }
    }
    fclose(f);

}
```

FIG. 2.

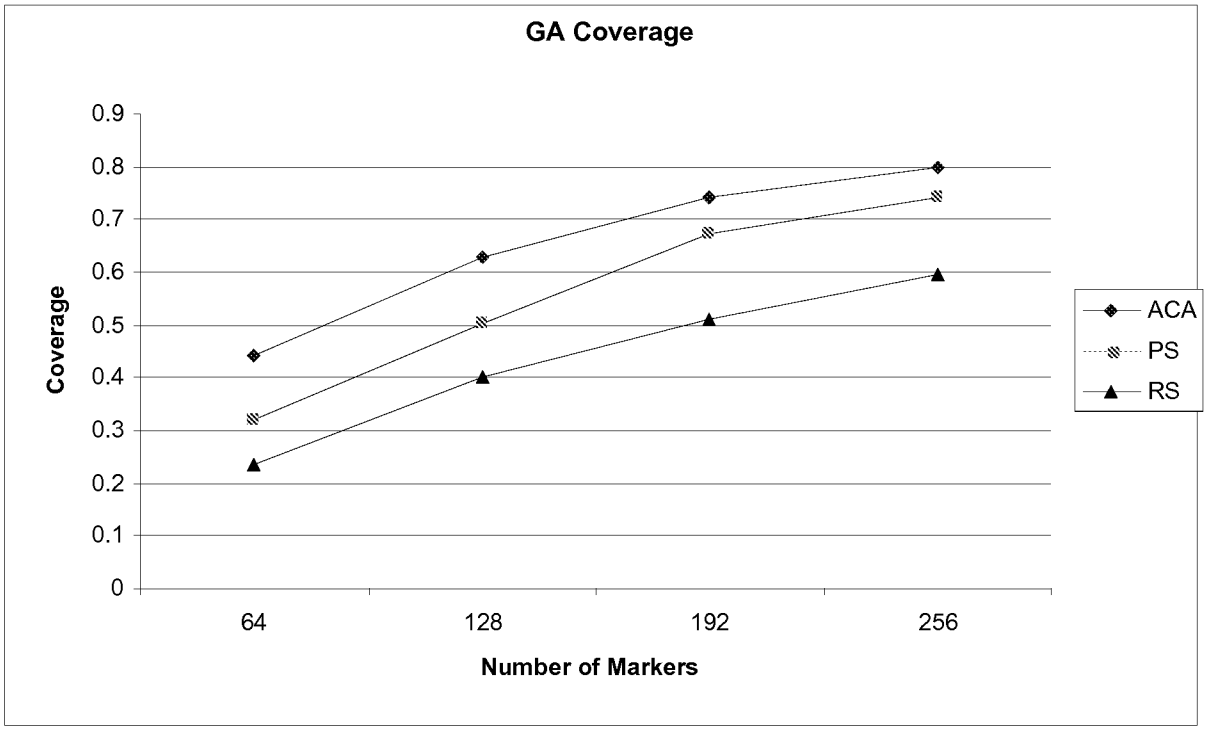

GA Coverage
Coverage
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
64
128
192
256
Number of Markers
ACA
PS
RS

FIG. 3.

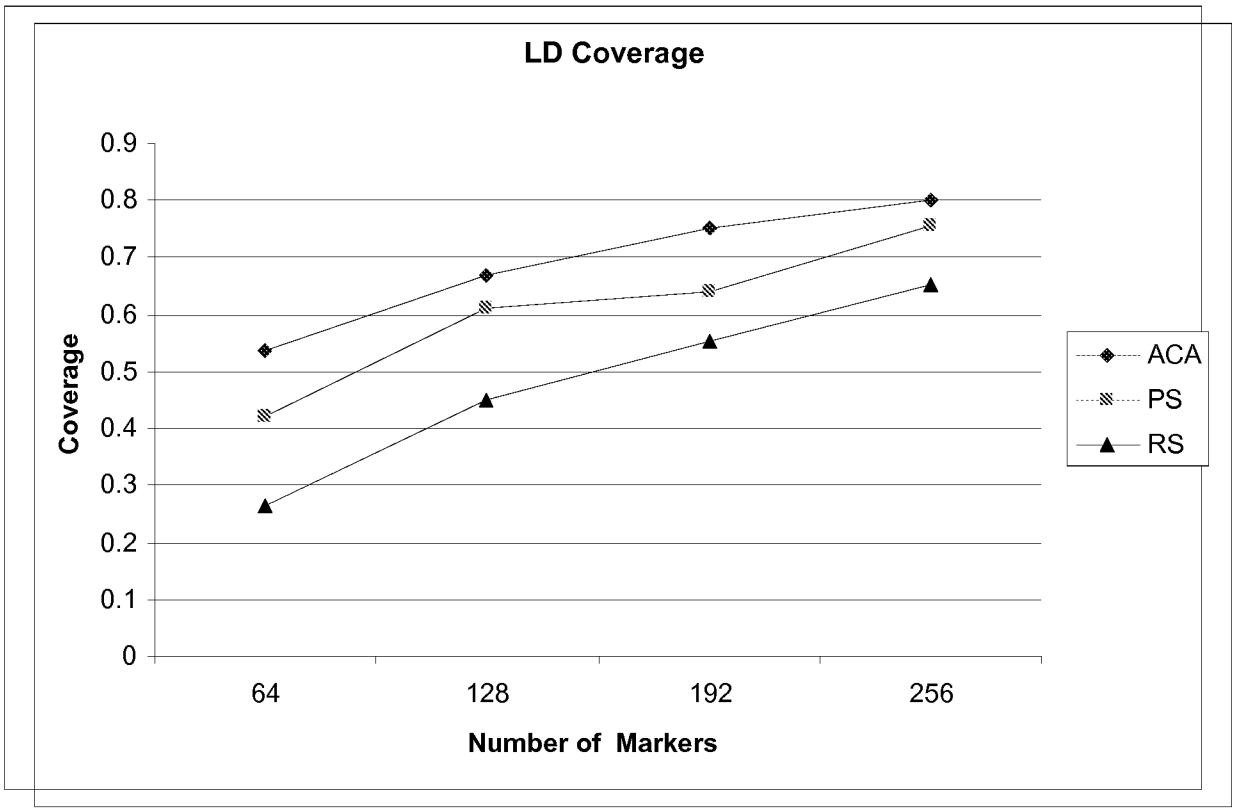
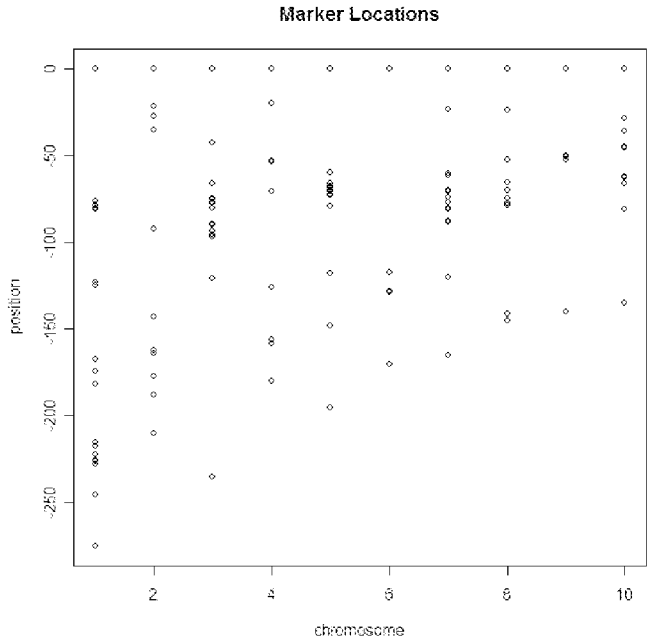

LD Coverage
Coverage
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
64
128
192
256
Number of Markers
ACA
PS
RS

FIG. 4a.

ACA Markers



All Markers

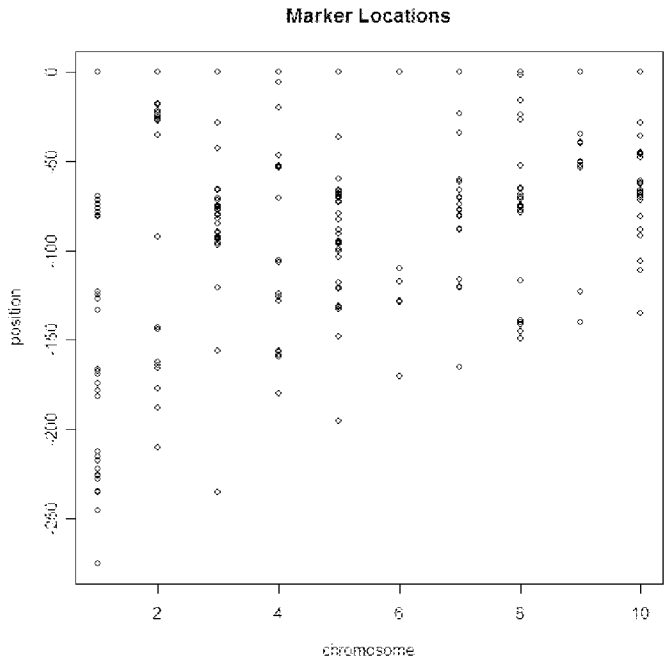
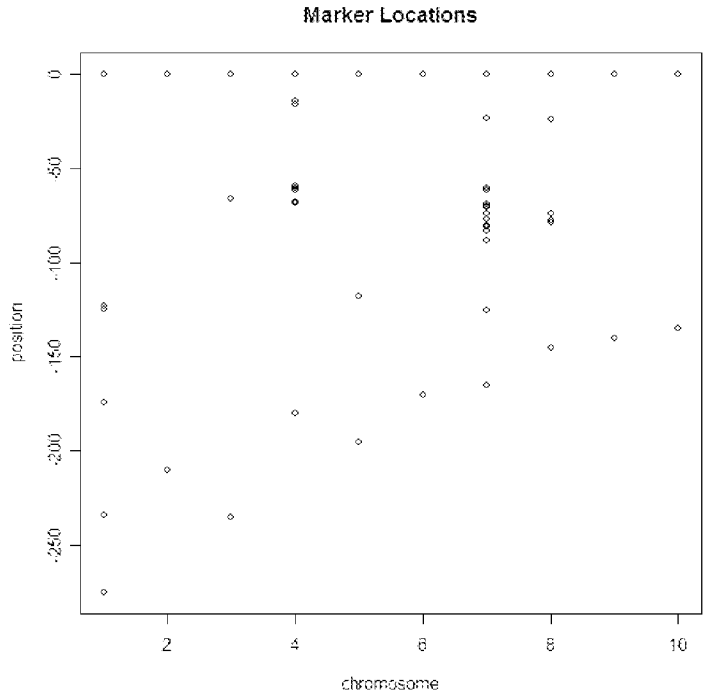


FIG. 4b.

ACA Markers



All Markers