Ant Colony Optimization

From Scholarpedia

From Scholarpedia, the free peer-reviewed encyclopedia p.18620

Curator: Marco Dorigo, IRIDIA, Université Libre de Bruxelles, Brussels, Belgium

Ant colony optimization (ACO) is a population-based metaheuristic that can be used to find approximate solutions to difficult optimization problems.

In ACO, a set of software agents called *artificial ants* search for good solutions to a given optimization problem. To apply ACO, the optimization problem is transformed into the problem of finding the best path on a weighted graph. The artificial ants (hereafter ants) incrementally build solutions by moving on the graph. The solution construction process is stochastic and is biased by a *pheromone model*, that is, a set of parameters associated with graph components (either nodes or edges) whose values are modified at runtime by the ants.

Contents

Explaining ACO through an Example

The easiest way to understand how ant colony optimization works is by means of an example. We consider its application to the traveling salesman problem (TSP). In the TSP a set of locations (cities) and the distances between them are given. The problem consists of finding a closed tour of minimal length that visits each city once and only once.

To apply ACO to the TSP, we consider the graph defined by associating the set of cities with the set of vertices of the graph. This graph is called *construction graph*. Since in the TSP it is possible to move from any given city to any other city, the construction graph is fully connected and the number of vertices is equal to the number of cities. We set the lengths of the edges between the vertices to be proportional to the distances between the cities represented by these vertices and we associate pheromone values and heuristic values with the edges of the graph. Pheromone values are modified at runtime and represent the cumulated experience of the ant colony, while heuristic values are problem dependent values that, in the case of the TSP, are set to be the inverse of the lengths of the edges.

The ants construct the solutions as follows. Each ant starts from a randomly selected city (vertex of the construction graph). Then, at each construction step it moves along the edges of the graph. Each ant keeps a memory of its path, and in subsequent steps it chooses among the edges that do not lead to vertices that it has already visited. An ant has constructed a solution once it has visited all the vertices of the graph. At each construction step, an ant probabilistically chooses the edge to follow among those that lead to yet unvisited vertices. The probabilistic rule is biased by pheromone values and heuristic information: the higher the pheromone and the heuristic value associated to an edge, the higher the probability an ant will choose that particular edge. Once all the ants have completed their tour, the pheromone on the edges is updated. Each of the pheromone values is initially decreased by a certain percentage. Each edge then receives an amount of additional pheromone proportional to the quality of the solutions to which it belongs (there is one solution per ant).

This procedure is repeatedly applied until a termination criterion is satisfied.

Formal Definition of a Combinatorial Optimization Problem

The first step for the application of ACO to a combinatorial optimization problem (COP) consists in defining a model of the COP as a triplet (S, Ω, f), where:

- S is a search space defined over a finite set of discrete decision variables;
- Ω is a set of constraints among the variables; and
- $f : S \rightarrow R_0^+$ is an objective function to be minimized (as maximizing over f is the same as minimizing over $-f$, every COP can be described as a minimization problem).

The search space S is defined as follows. A set of discrete variables X_i, $i = 1, \ldots, n$, with values $v_i^j \in D_i = \{v_i^1, \ldots, v_i^{|D_i|}\}$, is given. Elements of S are full assignments, that is, assignments in which each variable X_i has a value v_i^j assigned from its domain D_i. The set of feasible solutions S_Ω is given by the elements of S that satisfy all the constraints in the set Ω.

A solution $s^* \in S_\Omega$ is called a global optimum if and only if: $f(s^*) \leq f(s)\ \forall s \in S_\Omega$. The set of all globally optimal solutions is denoted by $S_\Omega^* \subseteq S_\Omega$. Solving a COP requires finding at least one $s^* \in S_\Omega^*$.

The Ant Colony Optimization Metaheuristic

In ACO, artificial ants build a solution to a combinatorial optimization problem by traversing a fully connected construction graph, defined as follows. First, each instantiated decision variable $X_i = v_i^j$ is called a *solution component* and denoted by c_{ij}. The set of all possible solution components is denoted by C. Then the construction graph $G_C(V,E)$ is defined by associating the components C either with the set of vertices V or with the set of edges E.

A pheromone trail value τ_{ij} is associated with each component c_{ij}. (Note that pheromone values are in general a function of the algorithm's iteration $t : \tau_{ij} = \tau_{ij}(t)$.) Pheromone values allow the probability distribution of different components of the solution to be modelled. Pheromone values are used and updated by the ACO algorithm during the search.

The ants move from vertex to vertex along the edges of the construction graph exploiting information provided by the pheromone values and in this way incrementally building a solution. Additionally, the ants deposit a certain amount of pheromone on the components, that is, either on the vertices or on the edges that they traverse. The amount $\Delta\tau$ of pheromone deposited may depend on the quality of the solution found. Subsequent ants utilize the pheromone information as a guide towards more promising regions of the search space.

The ACO metaheuristic is:

```
Set parameters, initialize pheromone trails
SCHEDULE_ACTIVITIES
  ConstructAntSolutions
  DaemonActions     {optional}
  UpdatePheromones
END_SCHEDULE_ACTIVITIES
```

The metaheuristic consists of an initialization step and of three algorithmic components whose activation is regulated by the Schedule_Activities construct. This construct is repeated until a termination criterion is met. Typical criteria are a maximum number of iterations or a maximum CPU time.

The Schedule_Activities construct does not specify how the three algorithmic components are scheduled and synchronized. In most applications of ACO to NP-hard problems however, the three algorithmic components undergo a loop that consists in (i) the construction of solutions by all ants, (ii) the (optional) improvement of these solution via the use of a local search algorithm, and (iii) the update of the pheromones. These three components are now explained in more details.

ConstructAntSolutions

A set of m artificial ants construct solutions from elements of a finite set of available solution components $C = \{c_{ij}\}$, $i = 1, \ldots, n$, $j = 1, \ldots, |D_i|$. A solution construction starts with an empty partial solution $s^p = \emptyset$. Then, at each construction step, the current partial solution s^p is extended by adding a feasible solution component from the set of feasible neighbors $N(s^p) \subseteq C$. The process of constructing solutions can be regarded as a path on the construction graph $G_C(V,E)$. The allowed paths in G_C are implicitly defined by the solution construction mechanism that defines the set $N(s^p)$ with respect to a partial solution s^p.

The choice of a solution component from $N(s^p)$ is done probabilistically at each construction step. The exact rules for the probabilistic choice of solution components vary across different ACO variants. The best known rule is the one of ant system (AS) (Dorigo et al. 1991, 1996):

$$p(c_{ij}|s^p) = \frac{\tau_{ij}^\alpha \cdot \eta_{ij}^\beta}{\sum_{c_{il} \in N(s^p)} \tau_{il}^\alpha \cdot \eta_{il}^\beta}, \forall c_{ij} \in N(s^p),$$

where τ_{ij} and η_{ij} are respectively the pheromone value and the heuristic value associated with the component c_{ij}. Furthermore, α

and β are positive real parameters whose values determine the relative importance of pheromone versus heuristic information.

DaemonActions

Once solutions have been constructed, and before updating the pheromone values, often some problem specific actions may be required. These are often called *daemon actions*, and can be used to implement problem specific and/or centralized actions, which cannot be performed by single ants. The most used daemon action consists in the application of local search to the constructed solutions: the locally optimized solutions are then used to decide which pheromone values to update.

UpdatePheromones

The aim of the pheromone update is to increase the pheromone values associated with good solutions, and to decrease those that are associated with bad ones. Usually, this is achieved (i) by decreasing all the pheromone values through *pheromone evaporation*, and (ii) by increasing the pheromone levels associated with a chosen set of good solutions S_{upd}:

$$\tau_{ij} \leftarrow (1-\rho)\cdot\tau_{ij} + \rho \cdot \sum_{s\in S_{upd}|c_{ij}\in s} F(s),$$

where S_{upd} is the set of solutions that are used for the update, $\rho \in (0,1]$ is a parameter called evaporation rate, and $F : S \rightarrow R_0^+$ is a function such that

$$f(s) < f(s') \Rightarrow F(s) \geq F(s'), \forall s \neq s' \in S .$$

$F(\cdot)$ is commonly called the *fitness function.*

Pheromone evaporation implements a useful form of *forgetting*, favoring the exploration of new areas in the search space. Different ACO algorithms, for example ant colony system (ACS) (Dorigo & Gambardella 1997) or MAX-MIN ant system (MMAS) (Stützle & Hoos 2000), differ in the way they update the pheromone.

Instantiations of the update rule given above are obtained by different specifications of S_{upd}, which in many cases is a subset of $S_{iter} \cup \{s_{bs}\}$, where S_{iter} is the set of solutions that were constructed in the current iteration, and s_{bs} is the *best-so-far* solution, that is, the best solution found since the first algorithm iteration. A well-known example is the AS-update rule, that is, the update rule of ant system (Dorigo et al. 1991, 1996): $S_{upd} \leftarrow S_{iter}$.

An example of a pheromone update rule that is more often used in practice is the IB-update rule (where IB stands for *iteration-best*):

$$S_{upd} \leftarrow \arg\max_{s\in S_{iter}} F(s).$$

The IB-update rule introduces a much stronger bias towards the good solutions found than the AS-update rule. Although this increases the speed with which good solutions are found, it also increases the probability of premature convergence. An even stronger bias is introduced by the BS-update rule, where BS refers to the use of the best-so-far solution s_{bs}. In this case, S_{upd} is set to $\{s_{sb}\}$. In practice, ACO algorithms that use variations of the IB-update or the BS-update rules and that additionally include mechanisms to avoid premature convergence, achieve better results than those that use the AS-update rule.

Main ACO Algorithms

Several special cases of the ACO metaheuristic have been proposed in the literature. Here we briefly overview, in the historical order in which they were introduced, the three most successful ones: ant system (Dorigo 1992, Dorigo et al. 1991, 1996), ant colony system (ACS) (Dorigo & Gambardella 1997), and MAX-MIN ant system (MMAS) (Stützle & Hoos 2000). In order to illustrate the differences between them clearly, we use the example of the traveling salesman problem.

Ant System

Ant system (AS) was the first ACO algorithm to be proposed in the literature (Dorigo et al. 1991, Dorigo 1992, Dorigo et al. 1996). Its main characteristic is that the pheromone values are updated by *all* the ants that have completed the tour. Solution components c_{ij} are the edges of the graph, and the pheromone update for τ_{ij} , that is, for the pheromone associated to the edge joining cities i and j , is performed as follows:

$$\tau_{ij} \leftarrow (1-\rho)\cdot\tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} ,$$

where $\rho \in (0,1]$ is the evaporation rate, m is the number of ants, and $\Delta\tau_{ij}^{k}$ is the quantity of pheromone laid on edge (i,j) by the k -th ant:

$$\Delta\tau_{ij}^{k} = \begin{cases} \frac{1}{L_k} & \text{if ant } k \text{ used edge } (i,j) \text{ in its tour,} \\ 0 & \text{otherwise,} \end{cases}$$

where L_k is the tour length of the k -th ant.

When constructing the solutions, the ants in AS traverse the construction graph and make a probabilistic decision at each vertex. The transition probability $p(c_{ij}|s_k^p)$ of the k -th ant moving from city i to city j is given by:

$$p(c_{ij}|s_k^p) = \begin{cases} \frac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum_{c_{il} \in N(s_k^p)} \tau_{il}^{\alpha} \cdot \eta_{il}^{\beta}} & \text{if } j \in N(s_k^p), \\ 0 & \text{otherwise,} \end{cases}$$

where $N(s_k^p)$ is the set of components that do not belong yet to the partial solution s_k^p of ant k , and α and β are parameters that control the relative importance of the pheromone versus the heuristic information $\eta_{ij} = 1/d_{ij}$, where d_{ij} is the length of component c_{ij} (i.e., of edge (i,j)).

Ant Colony System

The first major improvement over the original ant system to be proposed was ant colony system (ACS), introduced by Dorigo and Gambardella (1997). The first important difference between ACS and AS is the form of the decision rule used by the ants during the construction process. Ants in ACS use the so-called *pseudorandom proportional* rule: the probability for an ant to move from city i to city j depends on a random variable q uniformly distributed over $[0,1]$, and a parameter q_0; if $q \leq q_0$, then, among the feasible components, the component that maximizes the product $\tau_{il}\eta_{il}^{\beta}$ is chosen, otherwise the same equation as in AS is used.

This rather greedy rule, which favors exploitation of the pheromone information, is counterbalanced by the introduction of a diversifying component: the *local pheromone update*. The local pheromone update is performed by all ants after each construction step. Each ant applies it only to the last edge traversed:

$$\tau_{ij} = (1-\varphi) \cdot \tau_{ij} + \varphi \cdot \tau_0,$$

where $\varphi \in (0,1]$ is the pheromone decay coefficient, and τ_0 is the initial value of the pheromone.

The main goal of the local update is to diversify the search performed by subsequent ants during one iteration. In fact, decreasing the pheromone concentration on the edges as they are traversed during one iteration encourages subsequent ants to choose other edges and hence to produce different solutions. This makes less likely that several ants produce identical solutions during one iteration. Additionally, because of the local pheromone update in ACS, the minimum values of the pheromone are limited.

As in AS, also in ACS at the end of the construction process a pheromone update, called *offline* pheromone update, is performed.

ACS offline pheromone update is performed only by the best ant, that is, only edges that were visited by the best ant are updated, according to the equation:

$$\tau_{ij} \leftarrow (1-\rho) \cdot \tau_{ij} + \rho \cdot \Delta\tau_{ij}^{best}$$

where $\Delta\tau_{ij}^{best} = 1/L_{best}$ if the best ant used edge (i,j) in its tour, $\Delta\tau_{ij}^{best} = 0$ otherwise (L_{best} can be set to either the length of the best tour found in the current iteration---*iteration-best*, L_{ib}---or the best solution found since the start of the algorithm---*best-so-far*, L_{bs}).

It should be noted that most of the innovations introduced by ACS were introduced first in Ant-Q, a preliminary version of ACS by the same authors.

MAX-MIN Ant System

MAX-MIN ant system (MMAS) is another improvement, proposed by Stützle and Hoos (2000), over the original ant system idea. MMAS differs from AS in that (i) only the best ant adds pheromone trails, and (ii) the minimum and maximum values of the pheromone are explicitly limited (in AS and ACS these values are limited implicitly, that is, the value of the limits is a result of the algorithm working rather than a value set explicitly by the algorithm designer).

The pheromone update equation takes the following form:

$$\tau_{ij} \leftarrow (1-\rho) \cdot \tau_{ij} + \Delta\tau_{ij}^{best},$$

where $\Delta\tau_{ij}^{best} = 1/L_{best}$ if the best ant used edge (i,j) in its tour, $\Delta\tau_{ij}^{best} = 0$ otherwise, where L_{best} is the length of the tour of the best ant. As in ACS, L_{best} may be set (subject to the algorithm designer decision) either to L_{ib} or to L_{bs}, or to a combination of both.

The pheromone values are constrained between τ_{min} and τ_{max} by verifying, after they have been updated by the ants, that all pheromone values are within the imposed limits: τ_{ij} is set to τ_{max} if $\tau_{ij} > \tau_{max}$ and to τ_{min} if $\tau_{ij} < \tau_{min}$. It is important to note that the pheromone update equation of MMAS is applied, as it is the case for AS, to all the edges while in ACS it is applied only to the edges visited by the best ants.

The minimum value τ_{min} is most often experimentally chosen (however, some theory about how to define its value analytically has been developed in (Stützle & Hoos 2000)). The maximum value τ_{max} may be calculated analytically provided that the optimum ant tour length is known. In the case of the TSP, $\tau_{max} = 1/(\rho \cdot L^*)$, where L^* is the length of the optimal tour. If L^* is not known, it can be approximated by L_{bs}. It is also important to note that the initial value of the trails is set to τ_{max}, and that the algorithm is restarted when no improvement can be observed for a given number of iterations.

Applications of ACO and Current Trends

The initial applications of ACO were in the domain of NP-hard combinatorial optimization problems. The largest body of ACO research is still, not surprisingly, to be found in this area. The interested reader will find a rather complete overview of these applications in (Dorigo & Stützle 2004).

Another application that was considered early in the history of ACO is routing in telecommunication networks. A particularly successful example of ACO algorithm in this domain is AntNet (Di Caro & Dorigo 1998).

Current research in ACO algorithms is devoted both to the development of theoretical foundations and to the application of the metaheuristic to new challenging problems.

The development of theoretical foundation was started by Gutjahr, who was the first to prove convergence in probability of an ACO algorithm (Gutjahr 2000). An overview of theoretical results available for ACO can be found in (Dorigo & Blum 2005).

Concerning applications, the use of ACO for the solution of dynamic, multiobjective, stochastic, continuous and mixed-variable optimization problems is a current hot topic, as well as the creation of parallel implementations capable of taking advantage of the new available parallel hardware.

Many papers reporting on current research can be found in the proceedings of the ANTS conference or in the Swarm Intelligence journal (see External Links section below).

References

J.-L. Deneubourg, S. Aron, S. Goss, and J. M. Pasteels. The self-organizing exploratory pattern of the Argentine ant. *Journal of Insect Behavior*, 3:159–168, 1990.

G. Di Caro and M. Dorigo. AntNet: Distributed stigmergetic control for communications networks. *Journal of Artificial Intelligence Research*, 9:317–365, 1998.

M. Dorigo. Optimization, *Learning and Natural Algorithms (in Italian)*. PhD thesis, Dipartimento di Elettronica, Politecnico di Milano, Milan, Italy, 1992.

M. Dorigo and C. Blum. Ant colony optimization theory: A survey. *Theoretical Computer Science*, 344(2–3):243–278, 2005.

M. Dorigo and L. M. Gambardella. Ant Colony System: A cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1(1):53–66, 1997.

M. Dorigo, V. Maniezzo, and A. Colorni. *Positive feedback as a search strategy*. Technical Report 91-016, Dipartimento di Elettronica, Politecnico di Milano, Milan, Italy, 1991.

M. Dorigo, V. Maniezzo, and A. Colorni. Ant System: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics – Part B*, 26(1):29–41, 1996.

M. Dorigo and T. Stützle. *Ant Colony Optimization*. MIT Press, Cambridge, MA, 2004.

S. Goss, S. Aron, J.-L. Deneubourg, and J. M. Pasteels. Self-organized shortcuts in the Argentine ant. *Naturwissenschaften*, 76:579–581, 1989.

W. J. Gutjahr. A Graph-based Ant System and its convergence. *Future Generation Computer Systems*, 16(8):873–888, 2000.

T. Stützle and H. H. Hoos. MAX–MIN Ant System. *Future Generation Computer Systems*, 16(8):889–914, 2000.

Appendix -- The Natural Inspiration

The name *Ant Colony Optimization* was chosen to reflect its original inspiration: the foraging behavior of some ant species. In particular, it was inspired by the double-bridge experiment performed by Jean-Louis Deneubourg and colleagues. In this experiment it was shown that ants are able to find the shortest path to a food source by collectively exploiting pheromones they deposit on the ground while moving. Although ACO has grown to become a fully fledged algorithmic framework and now includes many components that are no longer related to real ants, we report here the double-bridge experiment for its historical value.

The Double-Bridge Experiment

In the *double bridge experiment*, a nest of a colony of *Argentine ants* is connected to a food source by two bridges. The ants can reach the food source and get back to the nest using any of the two bridges. The goal of the experiment is to observe the resulting behavior of the colony. What is observed is that if the two bridges have the same length, the ants tend to converge towards the use of one of the two bridges. If the experiment is repeated a number of times, it is observed that each of the two bridges is used in about 50% of the cases. These results can be explained by the fact that, while moving, ants deposit pheromone on the ground; and whenever they must choose which path to follow, their choice is biased by pheromone: the higher the pheromone concentration found on a particular path, the higher is the probability to follow that path.

Let us consider the case in which the two bridges have the same length. How the ants converge towards the use of a single bridge can be better understood with the help of Figure 1 .

At the start of the experiment the ants explore the surroundings of the nest. When they arrive at the decision point in which they have to choose which of the two bridges to use, they choose probabilistically, with a probability biased by the pheromone they sense on the two bridges. Initially, each ant chooses one of the two bridges with 50% probability as there is no pheromone yet. However, after some time, because of random fluctuations, one of the two bridges presents a higher concentration of pheromone than the other and, therefore, attracts more ants. This in turn increases the pheromone level on that bridge, making it more attractive. It is this autocatalytic mechanism that makes the whole colony converge towards the use of the same bridge.

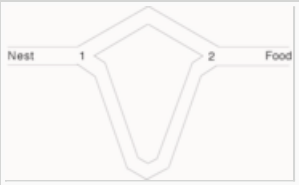

Figure 2: Fig.2 - Branches have different lengths (Goss et al 1989).

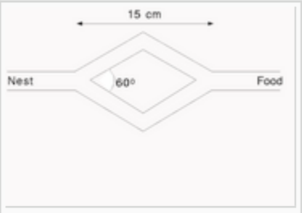

Figure 1: Fig.1 - Branches have equal lengths (Deneubourg et al. 1990).

If one of the bridges is significantly shorter than the other, a second mechanism plays an important role: the ants that happen randomly to choose the shorter bridge are the first to reach the food source. When these ants, while moving back to the nest, encounter the decision point 2 (see Figure 2), they sense a higher pheromone on the shorter bridge, which is then chosen with higher probability and once again receives additional pheromone. This fact increases the probability that further ants select it rather than the long one.

Goss et al. (1989) developed a model of the observed behavior: assuming that at a given moment in time, m_1 ants have used the first bridge and m_2 the second one, the probability p_1 for an ant to choose the first bridge is:

$$p_1 = \frac{(m_1 + k)^h}{(m_1 + k)^h + (m_2 + k)^h} \,,$$

where parameters k and h are to be fitted to the experimental data---obviously, $p_2 = 1 - p_1$. Monte Carlo simulations showed a very good fit for $k \approx 20$ and $h \approx 2$ (Goss et al. 1989).

It is this equation that inspired the equation used in ant system, the first ACO algorithm.

External Links

- The main reference about ACO is the book *Ant Colony Optimization* (http://mitpress.mit.edu/catalog/item/default.asp?ttype=2&tid=10139) .
- www.aco-metaheuristic.org (http://www.aco-metaheuristic.org/) : These are the official web pages dedicated to collect information about ACO.
- Software, distributed under the GNU license, is available at www.aco-metaheuristic.org/aco-code/ (http://iridia.ulb.ac.be/~mdorigo/ACO/aco-code/public-software.html)
- ACO is often covered by the popular press. Pointers to popularization articles can be found at: www.aco-metaheuristic.org/aco-in-the-press.html (http://iridia.ulb.ac.be/~mdorigo/ACO/aco-in-the-press.html)
- A moderated mailing list dedicated to the exchange of information related to ACO is accessible at: www.aco-metaheuristic.org/mailing-list.html (http://iridia.ulb.ac.be/~mdorigo/ACO/mailing-list.html)
- Many journals and conferences publish papers on ACO:
  - The main journal reporting research on ACO is Swarm Intelligence (http://www.springer.com/11721) . Other journals where papers on ACO regularly appear are Artificial Life, Evolutionary Computation, IEEE Transactions on Systems, Man, and Cybernetics, IEEE Transactions on Evolutionary Computation, INFORMS Journal on Computing, Journal of Operations Research Society, and European Journal of Operational Research.
  - *ANTS - From Ant Colonies to Artificial Ants: A Series of International Workshops on Ant Algorithms* (http://iridia.ulb.ac.be/~ants) . This biannual series of workshops, held for the first time in 1998, is the oldest conference in the ACO and swarm intelligence fields. Another more recent series of conferences dedicated to swarm intelligence are the annual IEEE Swarm Intelligence Symposia, started in 2003.
  - Special sessions or special tracks on ACO are organized in many conferences. Examples are the IEEE Congress on Evolutionary Computation (CEC) and the Genetic and Evolutionary Computation (GECCO) series of conferences.
  - Papers on ACO can regularly be found also in many other conferences such as Parallel Problem Solving from Nature conferences, INFORMS meetings, ECCO conferences, the Metaheuristics International Conference, the European Workshop on Evolutionary Computation in Combinatorial Optimization and many others.

See Also

Swarm Intelligence, Particle Swarm Optimization

Marco Dorigo (2007) Ant Colony Optimization. Scholarpedia, p.10721
Created: 2 May 2006, reviewed: 27 March 2007, accepted: 28 March 2007
Retrieved from "http://www.scholarpedia.org/article/Ant_Colony_Optimization"

Categories: Computational Intelligence | Artificial Intelligence