Counting authorised paths in constrained control-flow graphs

Enka Blanchard, Siargey Kachanovich

Counting authorised paths in constrained control-flow graphs

Enka Blanchard [1] and *Siargey Kachanovich* [2]

[1] IRIF, Université Paris Diderot, France & Loria, Université de Lorraine
[2] Université Côte d'Azur, INRIA Sophia-Antipolis, France

Extended Abstract

1 Introduction and problem definition

Our goal in this extended abstract is to investigate a model of computation inspired by control-flow graphs (CFG) [All70], automata and arithmetic circuits. The objective is to extend the definition of the first to include computing on nodes and edges.

We are given a directed graph and a set of integer variables, with every edge in the graph being augmented in two ways. First, a set of constraints on the variables determines whether the edge is *authorised*. Second, to each edge is associated a function that affects the values of the variables. In this complex model, we seek to estimate the length and number of distinct paths. We restrict ourselves to the two cases in which this has meaning: when the underlying graph is a DAG, or when we have a promise that paths of infinite length do not exist in the given graph. This has varied applications, from bounding the number of execution paths in various computing models (both deterministic and non-deterministic), to counting the number of distinct stories in interactive fictions. This problem is easy in the case of trees — checking whether each node is reachable can be done in linear time by a BFS — or in the case of directed acyclic graphs, where we can use dynamic programming. However, adding constraints on the values of variables can make it untractable, even with small graphs.

We focus on a simplified model — *constrained control-flow graphs* (CCFG) — defined by:

- A directed graph G with n nodes and $m \leq n^2$ directed edges (loops are authorised).
- A vector of k variables $V = (v_1, ..., v_k)$, all initialised at 0.
- A function f_e for each edge e, that associates a boolean to V, taking the value `TRUE` only if the edge is authorised for the value of V (see below).
- A vector V^e for each edge e, with the rule that, when going through e on a path, we assign $V := V + V_e$. This vector can have negative coordinates.
- An integer B, such that $\forall (V, e),\ \Big(f_e(V) \implies \forall i,\ (0 \leq v_i + V_i^e \leq B)\Big)$.

In such a model, we seek to count or estimate the maximum number Π of *authorised paths*, that is, paths such that each edge taken follows the constraints of f_e. We are also interested in the lengths of such paths, as they have a direct impact on the number of paths. The maximal length of any authorised path will be denoted by Λ.

Crucially, Π and Λ only exist if there is no path of infinite length. An equivalent property is that there exists no cycle that leaves all variables unchanged.

2 Upper bounds

We start by giving upper bounds on Λ. The first is useful when there are few variables, whereas the second addresses the case when the variables greatly outnumber the nodes.

Lemma 1. *All authorised paths have length bounded by* $n \times (B+1)^k$.

Proof. As each variable can take $B+1$ different values, there are $n \times (B+1)^k$ pairs (a, V) where a is a node. If a path has length $> n \times (B+1)^k$, one such pair must be present twice, and there is a cycle that leaves the variables unchanged. □

We can get an alternative upper bound on Λ in the case of $m < k$.

Lemma 2. *All authorised paths have length bounded by* $n \times 2^m k^{m/2} \binom{k}{m} B^m$.

Proof. We will prove the lemma geometrically by a volume argument. All possible values of the vector V are integer grid points inside a k-dimensional cube C of side B. The values V_e for all edges e span a linear space L of dimension at most m. Our goal is to estimate the number of integer grid points N that lie on the intersection $C \cap L$. Once we have this number, we can substitute the factor B^k in Lemma 1 to find a bound on the length of a path.

For each of the N points on $C \cap L$, we construct a ball of radius $1/2$. Because these points have integer coordinates, the constructed balls have disjoint interiors. All these balls lie in a tubular neighbourhood (thickening) S of $C \cap L$ of radius $1/2$. With $\mathrm{vol}(B_k)$ denoting the volume of a k-dimensional unit ball, we obtain: $N \times \mathrm{vol}(B_k)(\frac{1}{2})^k \leq \mathrm{vol}(S)$. The volume of S is given by Weyl's tube formula [Wey39]: $\mathrm{vol}(S) \in O\left(\mathrm{vol}(C \cap L)\,\mathrm{vol}(B_{k-m})\,(1/2)^{k-m}\right)$. By combining the two formulae, we get: $N \in O\left(2^m \mathrm{vol}(C \cap L) \frac{\mathrm{vol}(B_{k-m})}{\mathrm{vol}(B_k)}\right)$.

Moreover, $\mathrm{vol}(C \cap L)$ is upper-bounded by the volume of an m-dimensional ball of radius the diagonal of the cube C. This volume is $\mathrm{vol}(B_m)(\sqrt{k}B)^m$. The volume of a k-dimensional unit ball is given by [OLBC10]: $\mathrm{vol}(B_k) \sim \left(\frac{2\pi e}{k}\right)^{k/2}$. We therefore have:

$$N \in O\left(2^m k^{m/2}\, B^m \frac{\mathrm{vol}(B_m)\mathrm{vol}(B_{k-m})}{\mathrm{vol}(B_k)}\right) \subseteq O\left(2^m k^{m/2}\, B^m \binom{k}{m}\right).$$

□

We can derive an easy bound on Π from Lemma 2.

Corollary 3. *We have:*

$$\Pi \in n^{O\left(2^m k^{m/2} \binom{k}{m} B^m\right)}.$$

Proof. The number of ways to compose a path of length Λ from n nodes is n^Λ. We substitute Λ by the upper bound from Lemma 2. □

Remark 4. For some types of graphs, the upper bound on Π is much smaller. For example, the number of paths in a DAG with no variables is upper-bounded by $2^{n-1} - 1$. This bound comes from the complete graph on n nodes oriented in a non-cyclic manner.

3 Lower bounds

We now give constructive lower bounds of the maximum Λ and Π for fixed k or n, in two cases corresponding to Lemmas 1 and 2. Before this, we start by an introductory example on multigraphs.

Multigraph example. If we allow multigraphs, we have a matching bound for Lemma 1. Consider a single node with k loop edges $e_1, \ldots, e_k$. To each edge e_i we assign a vector $V^i = (-B, \ldots, -B, 1, 0, \ldots, 0)$, with 1 at the i-th place. Each path corresponds to the iterative incrementation of a counter in base $B+1$, from 0 to $(B+1)^k - 1$. We can also add a path of length $n-1$ that ends in node n without changing the variables. From node n, all the loops V^i are replaced by edges towards the first node. This gives: $\Lambda = n \times (B+1)^k$.

Small k case. If we get back to simple directed graphs — without parallel edges — we have a first lower bound on path length in the case $k < n$. We consider a graph $G_{n,k}$ derived from the previous one, where instead of the edge with vector V^i going from the n-th to the first node, it goes from the n-th to the i-th node. $G_{n,k}$ is then composed of a path on n nodes that does not affect the variables, and k edges from node n to the first k nodes of the path.

As before, the vector V evolves as an incremental integer counter with digits in base $B+1$ until it reaches the value $(B,\dots,B)$, returns to the last vertex, and stops as no edge is authorised. To increment the digit at a position i, the path follows the edge from n to i and then returns to n by passing through $i+1,\dots,n-1$. We now compute the total length of this path. The number of times the path passes through the edge from n to i is $B(B+1)^{k-i}$. Afterwards, the path returns to n by following $n-i$ edges. The length of the path is therefore:

$$\sum_{i=1}^{k} B(B+1)^{k-i}(n-i+1) = \frac{n(B+1)^{k+1} - (n+1)(B+1)^k - (n-k)B + 1}{B} \in \Theta(n(B+1)^k).$$

Small n case. When $n^2 < k$, we consider a second kind of graph. This graph consists of a complete graph K_{n-1} on $n-1$ nodes — including loops, hence with $(n-1)^2$ edges — and a distinguished node n connected to all other nodes in both ways. Each edge within the complete subgraph K_{n-1} has an associated unique vector of type $(-B,\dots,-B,1,0,\dots,0)$. The edges from n to other nodes are associated to the same vector $(1,0,\dots,0)$, while the edges to the node n are associated to the zero vector.

As was the case in the previous graph, the longest authorised path simulates an incremental integer counter on the vector V. This path starts at the node n. To increment the units digit, the path follows any edge from n to a node in K_{n-1}. If the next incrementation is in the digits of some position $i > 1$, then the chosen node is such that there is an outgoing edge associated to a vector $(-B,\dots,-B,1,0,\dots,0)$ with 1 at the i-th coordinate. In this case, the path follows this edge and returns to n. Otherwise, the path simply returns to n. The length of this path is therefore:

$$\Lambda = 2B(B+1)^{(n-1)^2-1} + (B+1)^{(n-1)^2-1} = (2B+1)(B+1)^{(n-1)^2-1}.$$

We can also count the number of maximal paths in this graph. For each incrementation of the units digit, the path can pass through any of the nodes in the complete subgraph K_{n-1} before returning to n. We obtain the value of Π:

$$\Pi = (n-1)^{(B-1)B^{(n-1)^2-1}}.$$

4 Using Markov chains to bound Π on given graphs

Algorithmically, finding the number of paths or the length of the greatest path in an arbitrary CCFG is a complex computational problem. The simplest way is to run a breadth-first search and count the number of endpoints, but this has a time complexity at least linear in Π. There are many potential practical improvements, such as analysis of DAG components and discarding redundant variables, but this does not help in the general case.

However, there is a way to give lower bounds on arbitrary graphs in practice. Intuitively, we create a Markov chain over the graph, which induces a probability distribution on all authorised paths, and use an estimator from a generalisation of the birthday problem [Mas92]. This allows us to get a time complexity in at most $\sqrt{\Pi}$ if the Markov chain has optimal values.

More formally, we give to each edge e a weight w_e, and we follow paths by taking a random edge with a distribution that depends on the weights of the authorised edges. Each time we stop (either because we reached a point with no outgoing edge or because the probability of outgoing edges was less than 1), we save that path. We keep drawing paths at random until we get one that we have saved previously. We output the square of the number of paths seen, which serves as an estimator for the total number of paths. For example, in the special case where all Π paths have an equal probability, we know that the expected number of trials T before a collision follows [CP+00]: $\frac{-2}{5} < T - \sqrt{\pi\Pi/2} < \frac{8}{5}$. As such, computing T^2 can give an estimate for Π. However, as the distribution is generally not uniform, T^2 is actually a lower bound for Π as the probability of getting a collision is a convex function of the p_i.

Divergence from the true value. Let us first assume that we have an optimal Markov chain to estimate the ratio $\frac{\Pi}{T^2}$, which increases with the non-uniformity of the distribution. We can then consider a simple graph with one node, one variable, and one loop that increments that variable. The corresponding Markov chain has a single parameter: the probability p of taking the loop edge. The path then has probability $(1-p) \times p^i$ to have length $i < B$, and probability p^B to have length B. Using Theorem 4 from [Wie05], a long but technically simple derivation shows that the optimal distribution to maximise T uses $p = \Theta\left((2n)^{\frac{-1}{2n}}\right)$. In such a case, $T \in \Theta(\sqrt{\frac{n}{\ln n}})$. Empirical values confirm the divergence, with $T \approx 9.26$ for $n = 100$, and $T \approx 23.7$ for $n = 1000$. This is for the simplest CCFG possible — a single node with a single loop — and a greater divergence might be found on more complex CCFGs.

Although there are limits on the efficiency of this method, it can still be of interest to find lower bounds. However, we have not yet solved the problem of finding the optimal Markov chain. A first possibility is to start with all $w_e = 1$ and repeatedly draw random paths until we get a collision. At this point, the w_e are incremented on all edges e of the colliding path. We then update the Markov chain, with the probability of taking each edge e being proportional to $\frac{1}{w_e}$. This iteratively diminishes the probability of getting that collision again, which tends to increase T. If the graph is a tree, a simple inductive proof shows that this process converges towards a uniform distribution on all paths. A cycle is not necessary to have divergence, as it can also occur on some DAGs.

5 Discussion and open problems

We have introduced a model of computation graphs and a set of bounds on the length and number of paths, as well as a method to estimate the latter. There are many open questions in this model, but the critical ones seem to be the following:

- What is the convergence speed of the Markov chain algorithm? Can it be improved by drawing paths until we have more than single collision before incrementing the weights?
- Can we obtain a useful upper bound on the divergence?
- Can we tighten the bounds in the small n case?

Acknowledgements. We wish to thank Sébastien Bouchard for fixing an error in the equation for the small n case's Λ. This work was supported partly by the french PIA project "Lorraine Université d'Excellence", reference ANR-15-IDEX-04-LUE.

References

[All70] Frances E Allen. Control flow analysis. In *ACM Sigplan Notices*, volume 5, pages 1–19. ACM, 1970.

[CP+00] Michael Camarri, Jim Pitman, et al. Limit distributions and random trees derived from the birthday problem with unequal probabilities. *Electronic Journal of Probability*, 5, 2000.

[Mas92] Shigeru Mase. Approximations to the birthday problem with unequal occurrence probabilities and their application to the surname problem in Japan. *Annals of the Institute of Statistical Mathematics*, 44(3):479–499, 1992.

[OLBC10] Frank W. J. Olver, Daniel W. Lozier, Ronald F. Boisvert, and Charles W. Clark. *NIST Handbook of Mathematical Functions*. Cambridge University Press, 2010.

[Wey39] Hermann Weyl. On the volume of tubes. *American Journal of Mathematics*, 61(2):461–472, 1939.

[Wie05] Michael J. Wiener. Bounds on birthday attack times. *IACR Cryptology ePrint Archive*, 2005:318, 2005.