

# The Homer System

Simon Colton<sup>1</sup> and Sophie Huczynska<sup>2</sup>

<sup>1</sup> Department of Computing, Imperial College, London, UK. [sgc@doc.ic.ac.uk](mailto:sgc@doc.ic.ac.uk)

<sup>2</sup> School of Informatics, Edinburgh University, UK. [shuczyns@inf.ed.ac.uk](mailto:shuczyns@inf.ed.ac.uk)

## 1 Introduction

The Homer system combines the HR automated theory formation program [2], the Otter theorem prover [6], and the Maple computer algebra package [1] to make intelligent conjectures about number theory functions supplied by the user. The integration is as follows: given Maple code for some functions the user is interested in, Maple is used to calculate values for those functions. HR then forms a theory using the functions as background knowledge, calling Maple whenever necessary to perform additional calculations. The theory formation process makes conjectures empirically and the user is initially asked to prove or disprove each conjecture. As the theory formation progresses, however, Homer uses the *theorems* it has found (namely those proved by the user) as axioms in attempts to prove the conjectures itself using Otter. Any conjectures proved in this way are likely to follow easily from the theorems already known to the user, so Homer does not present them, in order to keep the quality of the output high. Using Otter in this extreme way is not meant to indicate that Otter can only prove trivial theorems, nor that HR produces too many dull conjectures. Rather, we wish to emphasise the power of the combined system (Homer) at discovering interesting conjectures by generate (HR) and quick test (Otter).

HR forms a theory by inventing new concepts and making and proving (or disproving) conjectures about the concepts. Concept formation is via a set of 12 production rules which build a new concept out of one (or two) old ones [5]. HR then evaluates the concepts using a weighted sum of measures of interestingness, which drives a heuristic search: new concepts are built from the most interesting old concepts. HR makes conjectures empirically, using the examples of the concepts. For instance, if it notices that two concepts have exactly the same examples, it will make the conjecture that the definitions of the concepts are logically equivalent [3]. HR settles conjectures using third party provers (e.g., Otter) and model generators (e.g., MACE). HR has an existing interface with Maple and Otter [4], but has no interface for mathematicians. Homer fills this gap by accepting user input at run-time, providing additional interaction with Otter and Maple, presenting the user with the most interesting conjectures about their functions, and hiding the underlying processes at work. To apply the system to a user-driven search for number theory conjectures, various upgrades to HR were required, which we report here. In §2, we discuss the functionality implemented in HR and Homer to enable the user to interact with the system to improve the theory formed. We have also implemented various routines to increase the quality of the conjectures Homer produces, as discussed in §3. To demonstrate the effectiveness of these improvements, in §4 we describe a session undertaken by a number theorist (second author above) with Homer.

## 2 User Interaction

The front-end to the HR system alone has over 300 GUI objects the user can tweak to fine tune the theory formation it performs. For this reason, we chose to (i) set numerous defaults in HR (ii) implement a simpler front-end (Homer) and (iii) design the overall system in such a way as to keep to a minimum the effort required by a new user to understand and employ it. Firstly, we enabled Homer to take Maple code, extract the names and bodies of functions, and give this information to HR. Hence, users are not required to learn any new formalism. Secondly, to gather information required from the user, we use pop-up boxes asking explicit questions that users cannot ignore. We have kept the number of questions down to a bare minimum: to initialise Homer, the user must (i) tell it the name of a Maple file to read (ii) tell it which functions in the file to make conjectures about, and (iii) specify the integers which are to be used to make the conjectures empirically. After this initialisation stage, the only questions Homer asks are about the conjectures HR makes.

We chose to present the user with only the simplest conjectures (essentially Horn clause implications that concept P implies literal G). Also, we only output conjectures which are about number types, e.g., one integer type implies another type. For each conjecture, if possible, Homer also presents the user with 1 or 2 alternative, stronger, conjectures from which the alternative would follow. In particular, if P is only true of only a small number ( $< 5$ ) of the integers available, Homer presents the alternative that the conjecture is true because the concept is only ever true for those integers. For example, HR makes the conjecture that  $\tau(a) = a \Rightarrow isprime(\sigma(a))$  [where  $\tau(n)$  = number of divisors of  $n$  and  $\sigma(n)$  = sum of divisors]. Homer adds to this the (true) alternative that  $\tau(a) = a \Leftrightarrow (a = 1 \text{ or } a = 2)$ , from which the original conjecture follows. The user is asked to choose one alternative, and they should choose the stronger conjecture if true, as this will help prune more dull conjectures later. Homer also presents the user with an equivalence conjecture – if one is available in HR's theory – from which the original follows. For instance, when HR conjectures that  $isprime(n) \Rightarrow \tau(n) = 2$ , Homer retrieves and presents the stronger conjecture from HR that  $isprime(n) \Leftrightarrow \tau(n) = 2$ , which the user can choose to verify or not.

For each conjecture presented, in addition to the various strength conjectures on offer, Homer also allows the user to (a) state that the conjecture should remain open, in which case Homer displays the conjecture in a HTML text box, to remind the user of this open conjecture (b) provide a counterexample (c) ask HR to search between a range of numbers for one or more counterexamples and (d) offer a different, more general theorem – in Otter format – from which the original conjecture follows. Finally, at some stage, the user may want to stop interacting, and would rather the system continued to produce results without intervention. For this reason, there is a final option stating that Homer should continue autonomously. As it does, it collects and displays any results it cannot prove. It also highlights the set of integers each conjecture applies to, so the user can make more informed value judgements. The user can peruse the results at will without the need to stop Homer running.

### 3 Improving the Quality of Conjectures

At present – to keep the system simple to use – the user cannot prune/order conjectures using the plethora of measures of interestingness available in HR. Instead, as described below, we have attempted to keep the output of dull conjectures (which fit into the three categories below) to a minimum.

- **Instances of tautologies**

Conjectures like  $q(x) \rightarrow q(x)$  are trivial and dull instances of tautologies. HR ‘forbids’ certain theory formation steps to restrict the concepts and conjectures it produces, and also detects many tautologies using a syntactic analysis of the concepts in the conjecture. Any tautologies remaining are likely to be proved by Otter, hence Homer will not report them to the user. In saturation based provers such as Otter, there are already technical notions of redundancy, and we aim to use procedures such as tautology elimination to improve Homer.

- **Proved to follow easily from previous results**

As the user only provides Maple code, and no axioms whatsoever about their functions, it is impossible for Otter to prove any conjectures initially, e.g., it is infeasible to expect Homer to realise that  $isprime(n) \Leftrightarrow \tau(n) = 2$  follows from the definitions of the concepts when it starts with no number theory axioms. However, once it has identified such axioms, and the user has verified them, it is important for Homer not to present conjectures which can be proved using these axioms. For instance, given that  $isprime(n) \Leftrightarrow \tau(n) = 2$  has been verified by the user, Homer decides not to present results such as  $isprime(n) \Rightarrow isodd(\sigma(\tau(n)))$ . The latter is the kind of disappointing result that we wish to avoid – it looks to have potential, but follows from the axiom supplied. Such results are withheld from the user because they are proved by Otter. In this case, Homer passes Otter the above axiom and all the ground facts it knows about the user-supplied functions such as  $\sigma(2) = 3$  and  $isodd(3)$ , which, in this case, are exactly what is required to prove the disappointing theorem. Note that Otter is only allowed 10 seconds to prove a theorem, hence it may be that more difficult (and possibly more interesting theorems) which Otter can prove in a longer time are presented to the user. We have also implemented a naive subsumption checking mechanism which detects conjectures that unify with one already proved. These conjectures are withheld to avoid redundant and weak results being presented.

- **True only because of the limited applicability of a concept**

Conjectures stating the low applicability of concepts (e.g., the conjecture,  $C$ , that the *only* even prime is 2), can be interesting: for an example, see the appendix. However, once we know such applicability results, anything which is true just because of this property is likely to be dull. For instance, once the user verifies conjecture  $C$ , Homer withholds theorems such as: even primes,  $n$ , are equal to  $\tau(n)$ . Low applicability theorems are difficult to avoid initially, but Homer ensures that if the user chooses an applicability result instead of the conjecture presented to them (as described in §2), this is passed as an axiom to Otter in future. For example, if the user verifies that  $\tau(a) = a \Leftrightarrow (a = 1 \text{ or } a = 2)$ , Homer will discard conjectures like:  $\tau(\tau(a)) = \tau(a) \Leftrightarrow isprime(\sigma(\tau(a)))$ , which follows from the given applicability result, as  $\tau(a)$  must be 1 or 2.

## 4 Results from an Initial Session

HR has been used many times with number theoretic functions such as  $\tau$  and  $\sigma$ . The application we envisage for Homer is for the user to explore a function of interest in the context of already existing mathematics. Hence, in the session described here, we tried to invent such a scenario. In particular, the subject for this session – whose background is in number theory – was asked to use Homer to find some conjectures about Euler’s totient function,  $\phi(n)$ , which counts the number of integers less than and co-prime to  $n$ . This was both the first time that Homer had worked with this function, and the first time the user had worked with Homer. As background knowledge, Homer was given the concepts of odd, even, prime and square numbers, together with the  $\tau$ ,  $\sigma$  and  $\phi$  functions. It was instructed to work with the integers 1 to 50.

The system was used interactively for a period of approximately 4 hours. This included time spent by the user verifying or disproving conjectures, and indeed, Homer was active for only a fraction of that time. The system presented only 59 of the thousands of conjectures HR made, none of which were instances of tautologies. That is, for those which were true, this was because of the definitions of the functions, not the format of the conjecture. 38 conjectures have since been proved by the user, 4 have been shown to be false, and 17 remain open. Most of those which were trivially true from the function definitions (7 out of 38) occurred at the start of the session. As we hoped, the ease by which the user settled these results decreased as the theory formation advanced. In fact, all but 2 of the first 30 have been proved true, while 8 of the final 10 remain undecided. Of the 38 proved conjectures, there were 4 which were ‘number theoretically interesting’ to the user, with each requiring roughly 6 to 10 lines of proof. Two such theorems are given in the appendix. In general, the conjectures which remained open were those which Homer made on the basis of minimal empirical evidence, and as such, are likely to be false. HR was not used to generate counterexamples to break any of these conjectures, as this functionality was only added later.

## 5 Conclusions and Further Work

We believe it is significant that, in her first session, using Homer with a function never seen by a HR implementation ( $\phi$ ), our subject produced and proved four results she deemed to be interesting. In [2], we lament that only around 10% of the conjectures produced by that version of HR were interesting. Hence, we also believe it is an achievement that Homer presented only 59 of the thousands of conjectures HR produced, all of which required some knowledge of number theory to settle. The success of this session relied upon additional user-interaction functionality and an improvement in the quality of the conjectures. Much more testing will be needed before we can look at the hypothesis that Homer can be fruitfully employed by number theorists, which is our aim. Given the complexity of the results produced, it seems more likely that an application to recreational mathematics and/or to the setting of tutorial questions for students will be more fruitful in the short term, and we plan to pursue these applications.

## Acknowledgements

This research was supported by EPSRC grant GR/M98012. We would like to thank the reviewers for their helpful comments and suggestions.

## References

1. M Abell and J Braselton. *Maple V Handbook*. Academic Press, 1994.
2. S Colton. *Automated Theory Formation in Pure Mathematics*. Springer, 2002.
3. S Colton. The HR system for theorem generation. In *Proceedings of CADE-18, LNAI, 2392*. Springer Verlag, 2002.
4. S Colton. Making conjectures about Maple functions. In *Proceedings of AISC'02 and Calculemus 02, LNAI 2385*. Springer Verlag, 2002.
5. S Colton, A Bundy, and T Walsh. Automatic identification of mathematical concepts. In *Proceedings of the 17th ICML, 2000*.
6. W McCune. The OTTER user's guide. Technical Report ANL/90/9, Argonne National Laboratories, 1990.

## Appendix A

Amongst the first conjectures Homer made about Euler's totient function, were: " $\phi(n)$  odd  $\Rightarrow \phi(n) = 1$ " and " $\phi(n) = 1 \Rightarrow n = 1$  or  $2$ ". Combining these conjectures suggested the following result:

**Lemma** Let  $n \in \mathbb{N}$ . Then  $\phi(n)$  is even  $\Leftrightarrow n > 2$ .

*Proof* For  $n = 1$  or  $2$ ,  $\phi(n) = 1$ . Suppose  $n \geq 3$ . Write  $n = p_1^{\alpha_1} \dots p_s^{\alpha_s}$ , where  $s \geq 1$  and the  $p_i$  are distinct primes. Then  $\phi(n) = p_1^{\alpha_1-1}(p_1-1) \dots p_s^{\alpha_s-1}(p_s-1)$ . Since  $n \geq 3$ , then either  $n = 2^\alpha$  ( $\alpha \geq 2$ ), or  $p_k$  is an odd prime for some  $1 \leq k \leq s$ . In the first case,  $\phi(n) = 2^{\alpha-1}$  where  $\alpha - 1 \geq 1$ , while in the second,  $p_k - 1$  is even and divides  $\phi(n)$ . Thus in both cases  $2|\phi(n)$ .  $\square$

Another conjecture Homer made about Euler's totient function was " $\phi(n)$  even and square  $\Rightarrow \tau(n)$  is even". HR also pointed out that the set of integers between 1 and 50 which satisfy the left hand definition are  $\{5, 8, 10, 12, 17, 32, 34, 37, 40, 48\}$ . The user decided this set was sufficiently large to suggest that the conjecture may hold, but sufficiently small to suggest that the property is interesting. Since  $\phi(n)$  being even for  $n > 2$  is already known, the conjecture can be stated as:

**Theorem** Let  $n \in \mathbb{N}$ ,  $n > 2$ . Suppose  $\phi(n)$  is square. Then  $\tau(n)$  is even.

*Proof* We prove the contrapositive, namely that, for  $n(> 2) \in \mathbb{N}$ , if  $\tau(n)$  is odd then  $\phi(n)$  is a non-square. Write  $n = \prod_{i=1}^s p_i^{\alpha_i}$ , where  $s \geq 1$ ,  $p_1 < \dots < p_s$  and  $\alpha_i \geq 1$  for all  $1 \leq i \leq s$ . Assume that  $\tau(n)$  is odd. Then  $n$  must be a square, since  $\tau(n) = (\alpha_1 + 1) \dots (\alpha_s + 1)$  and so all the  $\alpha_i$  must be even. So, in fact,  $\alpha_i \geq 2$  for all  $i = 1, \dots, s$ . We have  $\phi(n) = \prod_{i=1}^s p_i^{\alpha_i-1}(p_i-1) = (p_1^{\alpha_1-1} \dots p_s^{\alpha_s-1})(p_1-1) \dots (p_s-1)$ . Thus  $p_s^{\alpha_s-1} || \phi(n)$  and, since  $\alpha_s - 1 (\geq 1)$  is odd,  $\phi(n)$  must be non-square.  $\square$

The user remarked that this result was "cute", and was interesting as it had to be viewed in the right way to be proved, i.e., by considering the contrapositive.