Tutorial on Separation Logic
(Invited Tutorial)

Peter O'Hearn*

Queen Mary, Univ. of London

Separation logic is an extension of Hoare's logic for reasoning about programs that manipulate pointers. Its assertion language extends classical logic with a separating conjunction operator $A * B$, which asserts that A and B hold for separate portions of memory.

In this tutorial I will first cover the basics of the logic, concentrating on highlights from the early work [1,2,3,4].

(i) The separating conjunction fits together with inductive definitions in a way that supports natural descriptions of mutable data structures [1].

(ii) Axiomatizations of pointer operations support *in-place reasoning*, where a portion of a formula is updated in place when passing from precondition to postcondition, mirroring the operational locality of heap update [1,2].

(iii) Notorious "dirty" features of low-level programming (pointer arithmetic, explicit deallocation) are dealt with smoothly, even embraced [2,3].

(iv) Frame axioms, which state what does not change, can be avoided when writing specifications [2,3].

These points together enable specifications and proofs of pointer programs that are dramatically simpler than was possible previously, in many cases approaching the simplicity associated with proofs of pure functional programs. I will describe how that is, and where rough edges lie (programs whose proofs are still more complex than we would like).

In describing these highlights I will outline how many of the points flow from Separation Logic's model theory, particularly an interaction between properties concerning the local way that imperative programs operate [5], and the abstract properties of its models, which it inherits from bunched logic [6,7] (a species of substructural logic related to linear and relevant logics, and Lambek's syntactic calculus). Using the model theoretic perspective, I will attempt to describe the extent to which Separation Logic's "benefits" do and do not depend on its language of assertions.

After the basic part, I will then discuss how these points (i)-(iv) feed into research on mechanized verification, both for interactive proof in proof assistants (e.g., [8,9,10,11,12]) and for automatic proof and abstract interpretation (e.g.

* Supported by the EPSRC and by a Royal Society Wolfson Research Merit Award.

A. Gupta and S. Malik (Eds.): CAV 2008, LNCS 5123, pp. 19–21, 2008.

[13,14,15,16,17,18,19,20,21,22,23]). Time permitting, I will close with more recent highlights, on concurrency, data abstraction, and object-oriented programming [24,25,26].

References

1. Reynolds, J.C.: Intuitionistic reasoning about shared mutable data structure. In: Davies, J., Roscoe, B., Woodcock, J. (eds.) Millennial Perspectives in Computer Science, Houndsmill, Hampshire, pp. 303–321. Palgrave (2000)
2. Isthiaq, S., O'Hearn, P.W.: BI as an assertion language for mutable data structures. In: 28th POPL, pp. 36–49 (2001)
3. O'Hearn, P., Reynolds, J., Yang, H.: Local reasoning about programs that alter data structures. In: Fribourg, L. (ed.) CSL 2001 and EACSL 2001. LNCS, vol. 2142, pp. 1–19. Springer, Heidelberg (2001)
4. Reynolds, J.C.: Separation logic: A logic for shared mutable data structures. In: 17th LICS, pp. 55–74 (2002)
5. Calcagno, C., O'Hearn, P., Yang, H.: Local action and abstract separation logic. In: 22nd LICS, pp. 366–378 (2007)
6. O'Hearn, P.W., Pym, D.J.: The logic of bunched implications. Bulletin of Symbolic Logic 5(2), 215–244 (1999)
7. Pym, D., O'Hearn, P., Yang, H.: Possible worlds and resources: the semantics of BI. Theoretical Computer Science 315(1), 257–305 (2004)
8. Yu, D., Hamid, N.A., Shao, Z.: Building certified libraries for PCC: Dynamic storage allocation. In: Degano, P. (ed.) ESOP 2003 and ETAPS 2003. LNCS, vol. 2618, pp. 363–379. Springer, Heidelberg (2003)
9. Marti, N., Affeldt, R., Yonezawa, A.: Formal verification of the heap manager of an operating system using separation logic. In: Liu, Z., He, J. (eds.) ICFEM 2006. LNCS, vol. 4260, pp. 400–419. Springer, Heidelberg (2006)
10. Tuch, H., Klein, G., Norrish, M.: Types, bytes, and separation logic. In: 34th POPL, pp. 97–108 (2007)
11. Myreen, M.O., Gordon, M.J.C.: Hoare logic for realistically modelled machine code. In: Grumberg, O., Huth, M. (eds.) TACAS 2007. LNCS, vol. 4424, pp. 568–582. Springer, Heidelberg (2007)
12. Varming, C., Birkedal, L.: Higher-order separation logic in Isabelle/HOLCF. In: 24th MFPS (2008)
13. Berdine, J., Calcagno, C., O'Hearn, P.W.: Smallfoot: Automatic modular assertion checking with separation logic. In: 4th FMCO, pp. 115–137 (2006)
14. Distefano, D., O'Hearn, P., Yang, H.: A Local Shape Analysis Based on Separation Logic. In: Hermanns, H., Palsberg, J. (eds.) TACAS 2006 and ETAPS 2006. LNCS, vol. 3920, pp. 287–302. Springer, Heidelberg (2006)
15. Magill, S., Nanevski, A., Clarke, E., Lee, P.: Inferring invariants in Separation Logic for imperative list-processing programs. In: 3rd SPACE Workshop (2006)
16. Berdine, J., Cook, B., Distefano, D., O'Hearn, P.: Automatic termination proofs for programs with shape-shifting heaps. In: Ball, T., Jones, R.B. (eds.) CAV 2006. LNCS, vol. 4144, pp. 386–400. Springer, Heidelberg (2006)
17. Gotsman, A., Berdine, J., Cook, B., Sagiv, M.: Thread-modular shape analysis. In: PLDI 2007 (2007)
18. Guo, B., Vachharajani, N., August, D.: Shape analysis with inductive recursion synthesis. In: PLDI (2007)

19. Chang, B., Rival, X., Necula, G.: Shape Analysis with Structural Invariant Checkers. In: Riis Nielson, H., Filé, G. (eds.) SAS 2007. LNCS, vol. 4634, pp. 384–401. Springer, Heidelberg (2007)
20. Berdine, J., Calcagno, C., Cook, B., Distefano, D., O'Hearn, P., Wies, T., Yang, H.: Shape analysis of composite data structures. In: Damm, W., Hermanns, H. (eds.) CAV 2007. LNCS, vol. 4590, pp. 178–192. Springer, Heidelberg (2007)
21. Nguyen, H.H., Chin, W.-N.: Enhancing program verification with lemmas. In: Gupta, A., Malik, S. (eds.) CAV 2008. LNCS, vol. 5123, pp. 355–369. Springer, Heidelberg (2008)
22. Magill, S., Tsai, M.-S., Lee, P., Tsay, Y.-K.: THOR: A tool for reasoning about shape and arithmetic. In: Gupta, A., Malik, S. (eds.) CAV 2008. LNCS, vol. 5123, pp. 428–432. Springer, Heidelberg (2008)
23. Yang, H., Lee, O., Berdine, J., Calcagno, C., Cook, B., Distefano, D., O'Hearn, P.: Scalable shape analysis for systems code. In: Gupta, A., Malik, S. (eds.) CAV 2008. LNCS, vol. 5123, pp. 385–398. Springer, Heidelberg (2008)
24. O'Hearn, P.W.: Resources, concurrency and local reasoning. Theoretical Computer Science 375(1-3), 271–307 (2007); Preliminary version appeared In: O'Hearn, P.W.: Resources, Concurrency and Local Reasoning. In: Gardner, P., Yoshida, N. (eds.) CONCUR 2004. LNCS, vol. 3170, pp. 49–67. Springer, Heidelberg (2004)
25. Parkinson, M., Bierman, G.: Separation logic and abstraction. In: 32nd POPL, pp. 59–70 (2005)
26. Biering, B., Birkedal, L., Torp-Smith, N.: BI-hyperdoctrines, higher-order separation logic, and abstraction. ACM TOPLAS 5(29) (2007)