Halting problem undecidability and infinitely nested simulation

When halting is defined as any computation that halts without ever having its simulation aborted then it can be understood that partial halt decider H correctly decides not halting on the simplified version of the Linz Ĥ. When this simplified concrete example is fully understood then the exact same reasoning can be applied to the actual Linz H correctly deciding not halting on its input.

The x86utm operating system was created so that the halting problem could be examined concretely in the high level language of C. x86utm UTM tape elements are 32-bit unsigned integers. H examines the behavior of the x86 emulation of its input. As soon as a non-halting behavior pattern is matched H aborts the simulation of its input and decides not halting.

```
int Simulate(u32 P, u32 I)
{
  ((void(*)(u32))P)(I);
  return 1;
}

// Simplified Linz Ĥ (Linz:1990:319)
void H_Hat(u32 P)
{
  // Linz H as a simulating partial halt decider
  u32 Input_Halts = H(P, P);
  if (Input_Halts)
    HERE: goto HERE;
}


void H_Hat2(u32 P)
{
  u32 Input_Halts = Simulate(P, P);
  if (Input_Halts)
    HERE: goto HERE;
}

int main()
{
  H_Hat2((u32)H_Hat2);
  H_Hat((u32)H_Hat);
}
```

Anyone that knows C programming very well will know that line 1 of main() won't halt and line 2 of main() will only halt if simulating partial halt decider H() stops simulating H_Hat(). A simulating halt decider that never stops simulating its input is simply a simulator on this input.

When we know that the UTM simulation of TM Description P on input I would never halt we know that the execution of TM P(I) would never halt.

On this basis we know that any computation that must have its simulation aborted to prevent its otherwise infinite execution is correctly rejected as non-halting.

Peter Linz Ĥ applied to the Turing machine description of itself: [Ĥ]

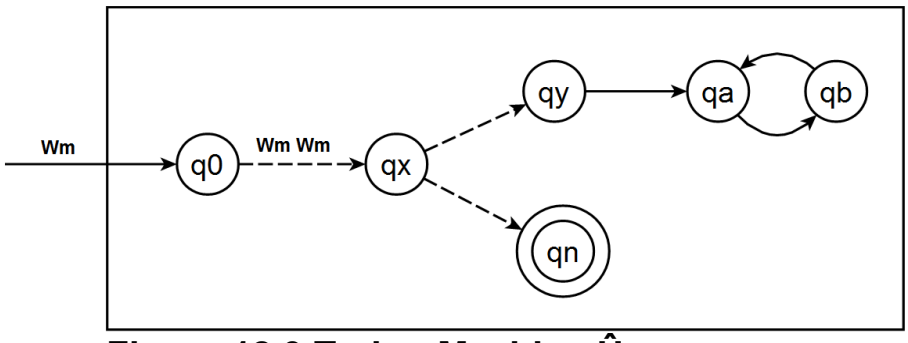

Figure 12.3 Turing Machine Ĥ

Ĥ.q0 wM ⊢* Ĥ.qx wM wM ⊢* Ĥ.qy ∞
Ĥ.q0 wM ⊢* Ĥ.qx wM wM ⊢* Ĥ.qn

The above is adapted from (Linz:1990:319).
It shows that Turing machine Ĥ copies its input at (q0) and begins executing an embedded copy of the original halt decider with this input at (qx).

The (qy) state indicates that the halt decider has determined that its input would halt. The ((qn)) state indicates the input would not halt. The appended (qa) and (qb) states cause Ĥ to infinitely loop if the halt decider decides that its input would halt.

The above definition specifies this execution trace:
It can be understood from the above specification that when the embedded halt decider at Ĥ.qx bases its halting decision on simulating its input, and it has ([Ĥ],[Ĥ]) as its input that:

- Ĥ.q0 would copy its input and then Ĥ.qx would simulate its input with this copy then
- Ĥ.q0 would copy its input and then Ĥ.qx would simulate its input with this copy then
- Ĥ.q0 would copy its input and then Ĥ.qx would simulate its input with this copy...

unless and until the halt decider at Ĥ.qx stops simulating its input.

When halting is defined as any computation that halts without ever having its simulation aborted by a simulating halt decider then we can see that the halt decider at state Ĥ.qx stops simulating its input and correctly transitions to its final Ĥ.qn state deciding not halting on its input in the computation: **Ĥ(⟨Ĥ⟩).**

When halting is defined as any computation that halts without ever having its simulation aborted then the fact that **Ĥ(⟨Ĥ⟩)** stops running does not indicate that it is a halting computation. The simulating halt decider would also force an infinite loop to stop running.

Linz, Peter 1990. An Introduction to Formal Languages and Automata. Lexington/Toronto: D. C. Heath and Company.

Appendix

```
int Factorial(int n)
{
  Output("Factorial(n)",n);
  if (n > 1)
    return n * Factorial(n - 1);
  else
    return 1;
}

void Infinite_Loop()
{
  HERE: goto HERE;
}


int Infinite_Recursion(u32 N)
{
  u32 M = Infinite_Recursion(N);
  return M;
}

void H_Hat(u32 P)
{
  u32 Input_Halts = H(P, P);
  if (Input_Halts)
    HERE: goto HERE;
}


int main()
{
  u32 Input_Would_Halt1 = H((u32)Factorial, 3);
  Output("Input_Would_Halt1 = ", Input_Would_Halt1);

  u32 Input_Would_Halt2 = H((u32)Infinite_Loop, (u32)Infinite_Loop);
  Output("Input_Would_Halt2 = ", Input_Would_Halt2);

  u32 Input_Would_Halt3 = H((u32)Infinite_Recursion, 3);
  Output("Input_Would_Halt3 = ", Input_Would_Halt2);

  u32 Input_Would_Halt4 = H((u32)H_Hat, (u32)H_Hat);
  Output("Input_Would_Halt4 = ", Input_Would_Halt4);
}
```

```
_Factorial()
[00000cc0](01)  55              push ebp
[00000cc1](02)  8bec            mov ebp,esp
[00000cc3](03)  8b4508          mov eax,[ebp+08]
[00000cc6](01)  50              push eax
[00000cc7](05)  68a3030000      push 000003a3
[00000ccc](05)  e85ff7ffff      call 00000430
[00000cd1](03)  83c408          add esp,+08
[00000cd4](04)  837d0801        cmp dword [ebp+08],+01
[00000cd8](02)  7e17            jng 00000cf1
[00000cda](03)  8b4d08          mov ecx,[ebp+08]
[00000cdd](03)  83e901          sub ecx,+01
[00000ce0](01)  51              push ecx
[00000ce1](05)  e8daffffff      call 00000cc0
[00000ce6](03)  83c404          add esp,+04
[00000ce9](04)  0faf4508        imul eax,[ebp+08]
[00000ced](02)  eb07            jmp 00000cf6
[00000cef](02)  eb05            jmp 00000cf6
[00000cf1](05)  b801000000      mov eax,00000001
[00000cf6](01)  5d              pop ebp
[00000cf7](01)  c3              ret
Size in bytes:(0056) [00000cf7]

_Infinite_Loop()
[00000d00](01)  55              push ebp
[00000d01](02)  8bec            mov ebp,esp
[00000d03](02)  ebfe            jmp 00000d03
[00000d05](01)  5d              pop ebp
[00000d06](01)  c3              ret
Size in bytes:(0007) [00000d06]

_Infinite_Recursion()
[00000d10](01)  55              push ebp
[00000d11](02)  8bec            mov ebp,esp
[00000d13](01)  51              push ecx
[00000d14](03)  8b4508          mov eax,[ebp+08]
[00000d17](01)  50              push eax
[00000d18](05)  e8f3ffffff      call 00000d10
[00000d1d](03)  83c404          add esp,+04
[00000d20](03)  8945fc          mov [ebp-04],eax
[00000d23](03)  8b45fc          mov eax,[ebp-04]
[00000d26](02)  8be5            mov esp,ebp
[00000d28](01)  5d              pop ebp
[00000d29](01)  c3              ret
Size in bytes:(0026) [00000d29]

_H_Hat()
[00000d30](01)  55              push ebp
[00000d31](02)  8bec            mov ebp,esp
[00000d33](01)  51              push ecx
[00000d34](03)  8b4508          mov eax,[ebp+08]
[00000d37](01)  50              push eax
[00000d38](03)  8b4d08          mov ecx,[ebp+08]
[00000d3b](01)  51              push ecx
[00000d3c](05)  e8dffdffff      call 00000b20
[00000d41](03)  83c408          add esp,+08
[00000d44](03)  8945fc          mov [ebp-04],eax
[00000d47](04)  837dfc00        cmp dword [ebp-04],+00
[00000d4b](02)  7402            jz 00000d4f
[00000d4d](02)  ebfe            jmp 00000d4d
[00000d4f](02)  8be5            mov esp,ebp
[00000d51](01)  5d              pop ebp
[00000d52](01)  c3              ret
Size in bytes:(0035) [00000d52]
```

```
_main()
[00000d60](01)  55              push ebp
[00000d61](02)  8bec            mov ebp,esp
[00000d63](03)  83ec10          sub esp,+10
[00000d66](02)  6a03            push +03
[00000d68](05)  68c00c0000      push 00000cc0
[00000d6d](05)  e8aefdffff      call 00000b20
[00000d72](03)  83c408          add esp,+08
[00000d75](03)  8945f8          mov [ebp-08],eax
[00000d78](03)  8b45f8          mov eax,[ebp-08]
[00000d7b](01)  50              push eax
[00000d7c](05)  68b3030000      push 000003b3
[00000d81](05)  e8aaf6ffff      call 00000430
[00000d86](03)  83c408          add esp,+08
[00000d89](05)  68000d0000      push 00000d00
[00000d8e](05)  68000d0000      push 00000d00
[00000d93](05)  e888fdffff      call 00000b20
[00000d98](03)  83c408          add esp,+08
[00000d9b](03)  8945fc          mov [ebp-04],eax
[00000d9e](03)  8b4dfc          mov ecx,[ebp-04]
[00000da1](01)  51              push ecx
[00000da2](05)  68cb030000      push 000003cb
[00000da7](05)  e884f6ffff      call 00000430
[00000dac](03)  83c408          add esp,+08
[00000daf](02)  6a03            push +03
[00000db1](05)  68100d0000      push 00000d10
[00000db6](05)  e865fdffff      call 00000b20
[00000dbb](03)  83c408          add esp,+08
[00000dbe](03)  8945f0          mov [ebp-10],eax
[00000dc1](03)  8b55fc          mov edx,[ebp-04]
[00000dc4](01)  52              push edx
[00000dc5](05)  68e3030000      push 000003e3
[00000dca](05)  e861f6ffff      call 00000430
[00000dcf](03)  83c408          add esp,+08
[00000dd2](05)  68300d0000      push 00000d30
[00000dd7](05)  68300d0000      push 00000d30
[00000ddc](05)  e83ffdffff      call 00000b20
[00000de1](03)  83c408          add esp,+08
[00000de4](03)  8945f4          mov [ebp-0c],eax
[00000de7](03)  8b45f4          mov eax,[ebp-0c]
[00000dea](01)  50              push eax
[00000deb](05)  68fb030000      push 000003fb
[00000df0](05)  e83bf6ffff      call 00000430
[00000df5](03)  83c408          add esp,+08
[00000df8](02)  33c0            xor eax,eax
[00000dfa](02)  8be5            mov esp,ebp
[00000dfc](01)  5d              pop ebp
[00000dfd](01)  c3              ret
Size in bytes:(0158) [00000dfd]

================================
...[00000d60][001018f3][00000000](01)  55              push ebp
...[00000d61][001018f3][00000000](02)  8bec            mov ebp,esp
...[00000d63][001018e3][90909090](03)  83ec10          sub esp,+10
...[00000d66][001018df][00000003](02)  6a03            push +03
...[00000d68][001018db][00000cc0](05)  68c00c0000      push 00000cc0
...[00000d6d][001018d7][00000d72](05)  e8aefdffff      call 00000b20
Begin Local Halt Decider Simulation at Machine Address:cc0
...[00000cc0][00211993][00211997](01)  55              push ebp
...[00000cc1][00211993][00211997](02)  8bec            mov ebp,esp
...[00000cc3][00211993][00211997](03)  8b4508          mov eax,[ebp+08]
...[00000cc6][0021198f][00000003](01)  50              push eax
...[00000cc7][0021198b][000003a3](05)  68a3030000      push 000003a3
---[00000ccc][0021198b][000003a3](05)  e85ff7ffff      call 00000430
Factorial(n)3
...[00000cd1][00211993][00211997](03)  83c408          add esp,+08
```

```
...[00000cd4][00211993][00211997](04)  837d0801        cmp dword [ebp+08],+01
...[00000cd8][00211993][00211997](02)  7e17            jng 00000cf1
...[00000cda][00211993][00211997](03)  8b4d08          mov ecx,[ebp+08]
...[00000cdd][00211993][00211997](03)  83e901          sub ecx,+01
...[00000ce0][0021198f][00000002](01)  51              push ecx
...[00000ce1][0021198b][00000ce6](05)  e8daffffff      call 00000cc0
...[00000cc0][00211987][00211993](01)  55              push ebp
...[00000cc1][00211987][00211993](02)  8bec            mov ebp,esp
...[00000cc3][00211987][00211993](03)  8b4508          mov eax,[ebp+08]
...[00000cc6][00211983][00000002](01)  50              push eax
...[00000cc7][0021197f][000003a3](05)  68a3030000      push 000003a3
---[00000ccc][0021197f][000003a3](05)  e85ff7ffff      call 00000430
Factorial(n)2
...[00000cd1][00211987][00211993](03)  83c408          add esp,+08
...[00000cd4][00211987][00211993](04)  837d0801        cmp dword [ebp+08],+01
...[00000cd8][00211987][00211993](02)  7e17            jng 00000cf1
...[00000cda][00211987][00211993](03)  8b4d08          mov ecx,[ebp+08]
...[00000cdd][00211987][00211993](03)  83e901          sub ecx,+01
...[00000ce0][00211983][00000001](01)  51              push ecx
...[00000ce1][0021197f][00000ce6](05)  e8daffffff      call 00000cc0
...[00000cc0][0021197b][00211987](01)  55              push ebp
...[00000cc1][0021197b][00211987](02)  8bec            mov ebp,esp
...[00000cc3][0021197b][00211987](03)  8b4508          mov eax,[ebp+08]
...[00000cc6][00211977][00000001](01)  50              push eax
...[00000cc7][00211973][000003a3](05)  68a3030000      push 000003a3
---[00000ccc][00211973][000003a3](05)  e85ff7ffff      call 00000430
Factorial(n)1
...[00000cd1][0021197b][00211987](03)  83c408          add esp,+08
...[00000cd4][0021197b][00211987](04)  837d0801        cmp dword [ebp+08],+01
...[00000cd8][0021197b][00211987](02)  7e17            jng 00000cf1
...[00000cf1][0021197b][00211987](05)  b801000000      mov eax,00000001
...[00000cf6][0021197f][00000ce6](01)  5d              pop ebp
...[00000cf7][00211983][00000001](01)  c3              ret
...[00000d72][001018e3][90909090](03)  83c408          add esp,+08
...[00000d75][001018e3][90909090](03)  8945f8          mov [ebp-08],eax
...[00000d78][001018e3][90909090](03)  8b45f8          mov eax,[ebp-08]
...[00000d7b][001018df][00000001](01)  50              push eax
...[00000d7c][001018db][000003b3](05)  68b3030000      push 000003b3
---[00000d81][001018db][000003b3](05)  e8aaf6ffff      call 00000430
Input_Would_Halt1 = 1
...[00000d86][001018e3][90909090](03)  83c408          add esp,+08
...[00000d89][001018df][00000d00](05)  68000d0000      push 00000d00
...[00000d8e][001018db][00000d00](05)  68000d0000      push 00000d00
...[00000d93][001018d7][00000d98](05)  e888fdffff      call 00000b20
Begin Local Halt Decider Simulation at Machine Address:d00
...[00000d00][0025c3bb][0025c3bf](01)  55              push ebp
...[00000d01][0025c3bb][0025c3bf](02)  8bec            mov ebp,esp
...[00000d03][0025c3bb][0025c3bf](02)  ebfe            jmp 00000d03
...[00000d03][0025c3bb][0025c3bf](02)  ebfe            jmp 00000d03
Local Halt Decider: Infinite Loop Detected Simulation Stopped
...[00000d98][001018e3][90909090](03)  83c408          add esp,+08
...[00000d9b][001018e3][90909090](03)  8945fc          mov [ebp-04],eax
...[00000d9e][001018e3][90909090](03)  8b4dfc          mov ecx,[ebp-04]
...[00000da1][001018df][00000000](01)  51              push ecx
...[00000da2][001018db][000003cb](05)  68cb030000      push 000003cb
---[00000da7][001018db][000003cb](05)  e884f6ffff      call 00000430
Input_Would_Halt2 = 0
...[00000dac][001018e3][90909090](03)  83c408          add esp,+08
...[00000daf][001018df][00000003](02)  6a03            push +03
...[00000db1][001018db][00000d10](05)  68100d0000      push 00000d10
...[00000db6][001018d7][00000dbb](05)  e865fdffff      call 00000b20
Begin Local Halt Decider Simulation at Machine Address:d10
...[00000d10][002a6de3][002a6de7](01)  55              push ebp
...[00000d11][002a6de3][002a6de7](02)  8bec            mov ebp,esp
...[00000d13][002a6ddf][00296db3](01)  51              push ecx
...[00000d14][002a6ddf][00296db3](03)  8b4508          mov eax,[ebp+08]
```

```
...[00000d17][002a6ddb][00000003](01)  50              push eax
...[00000d18][002a6dd7][00000d1d](05)  e8f3ffffff      call 00000d10
...[00000d10][002a6dd3][002a6de3](01)  55              push ebp
...[00000d11][002a6dd3][002a6de3](02)  8bec            mov ebp,esp
...[00000d13][002a6dcf][00296db3](01)  51              push ecx
...[00000d14][002a6dcf][00296db3](03)  8b4508          mov eax,[ebp+08]
...[00000d17][002a6dcb][00000003](01)  50              push eax
...[00000d18][002a6dc7][00000d1d](05)  e8f3ffffff      call 00000d10
Local Halt Decider: Infinite Recursion Detected Simulation Stopped
...[00000dbb][001018e3][90909090](03)  83c408          add esp,+08
...[00000dbe][001018e3][00000000](03)  8945f0          mov [ebp-10],eax
...[00000dc1][001018e3][00000000](03)  8b55fc          mov edx,[ebp-04]
...[00000dc4][001018df][00000000](01)  52              push edx
...[00000dc5][001018db][000003e3](05)  68e3030000      push 000003e3
---[00000dca][001018db][000003e3](05)  e861f6ffff      call 00000430
Input_Would_Halt3 = 0
...[00000dcf][001018e3][00000000](03)  83c408          add esp,+08
...[00000dd2][001018df][00000d30](05)  68300d0000      push 00000d30
...[00000dd7][001018db][00000d30](05)  68300d0000      push 00000d30
...[00000ddc][001018d7][00000de1](05)  e83ffdffff      call 00000b20
Begin Local Halt Decider Simulation at Machine Address:d30
...[00000d30][002f180b][002f180f](01)  55              push ebp
...[00000d31][002f180b][002f180f](02)  8bec            mov ebp,esp
...[00000d33][002f1807][002e17db](01)  51              push ecx
...[00000d34][002f1807][002e17db](03)  8b4508          mov eax,[ebp+08]
...[00000d37][002f1803][00000d30](01)  50              push eax
...[00000d38][002f1803][00000d30](03)  8b4d08          mov ecx,[ebp+08]
...[00000d3b][002f17ff][00000d30](01)  51              push ecx
...[00000d3c][002f17fb][00000d41](05)  e8dffdffff      call 00000b20
...[00000d30][0033c233][0033c237](01)  55              push ebp
...[00000d31][0033c233][0033c237](02)  8bec            mov ebp,esp
...[00000d33][0033c22f][0032c203](01)  51              push ecx
...[00000d34][0033c22f][0032c203](03)  8b4508          mov eax,[ebp+08]
...[00000d37][0033c22b][00000d30](01)  50              push eax
...[00000d38][0033c22b][00000d30](03)  8b4d08          mov ecx,[ebp+08]
...[00000d3b][0033c227][00000d30](01)  51              push ecx
...[00000d3c][0033c223][00000d41](05)  e8dffdffff      call 00000b20
Local Halt Decider: Infinite Recursion Detected Simulation Stopped
...[00000de1][001018e3][00000000](03)  83c408          add esp,+08
...[00000de4][001018e3][00000000](03)  8945f4          mov [ebp-0c],eax
...[00000de7][001018e3][00000000](03)  8b45f4          mov eax,[ebp-0c]
...[00000dea][001018df][00000000](01)  50              push eax
...[00000deb][001018db][000003fb](05)  68fb030000      push 000003fb
---[00000df0][001018db][000003fb](05)  e83bf6ffff      call 00000430
Input_Would_Halt4 = 0
...[00000df5][001018e3][00000000](03)  83c408          add esp,+08
...[00000df8][001018e3][00000000](02)  33c0            xor eax,eax
...[00000dfa][001018f3][00000000](02)  8be5            mov esp,ebp
...[00000dfc][001018f7][00100000](01)  5d              pop ebp
...[00000dfd][001018fb][000001d0](01)  c3              ret
Number_of_User_Instructions(117)
Number of Instructions Executed(32231)
```