EARLY DRAFT Proposal for reversers based on Forwarders as described in:
http://mail.openjdk.java.net/pipermail/valhalla-spec-experts/2019-April/000912.html

This is very early days - need to explore many more examples.

Rule 1: if override forwarder and do not locally override forwardee target (even if super overrides forwardee target): need reverser, so that your local implementation will override the updated method

1a. This needs to detect if indirect override forwarder
- even if my super created a reverser, my reverser may override that reverser

1b. Even if you override a forwarder with a forwarder, you still need to create the reverser if there is no local override for the target.

1c. If you override a forwarder with a forwarder, and there is a local forwarder that overrides the target, if the reverser would not form a loop, you still need to create the reverser.

Colors:
new source
new class file forwarder
new vm reverser

Assumption: assume recompilation leaves class file unchanged for the exercises.

EXAMPLE I:

Old:
D.m(Date)
E extends D { E.m(Date) }
F extends E { F.m(Date) }

Migration step 1: source migration D.m(Date) -> D.m(LDT)
source: D.m(Date) -> D.m(LDT)
class file:
D.m(LDT) {}
D.m(Date) FW-> D.m(LDT)
E.m(Date)
F.m(Date)

Runtime reversers created:
E.m(LDT) -> E.m(Date) // E.m(Date) overrides D.m(Date)FW: create reverser
F.m(LDT) -> F.m(Date) // F.m(Date) @overrides D.m(Date)FW: create reverser

invocation	dynamic receiver	resolution NOT invoked	selection: actual execution
invokevirtual D::m(Date)	D	D.m(Date)	D.m(Date) forwarder: adapt Date->LDT invoke local D.m(LDT) if return had changed, adapt
	E	“”	E.m(Date) // original - unchanged behavior
	F	“”	F.m(Date) // original - unchanged behavior
invokevirtual E.m(Date)	E	E.m(Date)	E.m(Date) // original - unchanged behavior
	F	“”	F.m(Date) // original - unchanged behavior
invokevirtual F.m(Date)	F	F.m(Date)	F.m(Date) // original - unchanged behavior
invokevirtual D::m(LDT)	D	D.m(LDT)	D.m(LDT)

	E	“”	E.m(LDT) // E.m(Date) overrode D.m(Date) FW
			reverser: adapt LDT->Date
			invoke local E.m(Date)
			if return had changed, adapt return back

	F	“”	F.m(LDT) // F.m(Date) @overrode D.m(Date) FW
			reverser: adapt LDT->Date
			invoke local F.m(Date)
			if return had changed, adapt return back

invokevirtual E.m(LDT)	E	E.m(LDT)
		reverser
			E.m(LDT): // E.m(Date) overrode D.m(Date) FW
			reverser: adapt LDT->Date
			invoke local E.m(Date)
			if return had changed, adapt return back

	F	F.m(LDT)
		reverser
			F.m(LDT) // F.m(Date) @overrode D.m(Date) FW
			reverser: adapt LDT->Date
			invoke local F.m(Date)
			if return had changed, adapt return back

invokevirtual F.m(LDT)	F	F.m(LDT)
		reverser
			F.m(LDT) // F.m(Date) @overrode D.m(Date) FW
			reverser: adapt LDT->Date
			invoke local F.m(Date)
			if return had changed, adapt return back

====
Migration step 2: source migration E.m(Date) -> E.m(LDT)
  source:
    D.m(Date) -> D.m(LDT)
    E.m(Date) -> E.m(LDT)
  class file:
    D.m(LDT) {}
    D.m(Date) FW -> D.m(LDT)
    E.m(LDT) {}

E.m(Date) FW -> E.m(LDT)
F.m(Date)

Runtime reversers created:
E.m(Date)FW overrides D.m(Date)FW AND local target E.m(LDT) overrides forwardee D.m(LDT): do NOT create reverser
F.m(LDT) -> F.m(Date) // F.m(Date) overrides E.m(Date)FW: create reverser

invocation	**dynamic receiver**	**resolution NOT invoked**	**selection: actual execution**
invokevirtual D::m(Date)	D	D.m(Date)	D.m(Date) forwarder: adapt Date->LDT invoke local D.m(LDT) if return had changed, adapt
	E	“”	E.m(Date) forwarder: adapt Date -> LDT invoke local E.m(LDT) if return had changed, adapt
	F	“”	F.m(Date) // original - unchanged behavior
invokevirtual E.m(Date)	E	E.m(Date)	E.m(Date) forwarder: adapt Date -> LDT invoke local E.m(LDT) if return had changed, adapt
	F	“”	F.m(Date) // original - unchanged behavior
invokevirtual F.m(Date)	F	F.m(Date)	F.m(Date) // original - unchanged behavior
invokevirtual D::m(LDT)	D	D.m(LDT)	D.m(LDT)
	E	“”	E.m(LDT)

	F	“”	F.m(LDT) // F.m(Date) overrode E.m(Date) FW
			reverser: adapt LDT->Date
			invoke local F.m(Date)
			if return had changed, adapt return back

invokevirtual E.m(LDT)	E	E.m(LDT)	E.m(LDT)

	F	“”	F.m(LDT) // F.m(Date) overrode E.m(Date) FW
			reverser: adapt LDT->Date
			invoke local F.m(Date)
			if return had changed, adapt return back

invokevirtual F.m(LDT)	F	F.m(LDT)
		reverser
			F.m(LDT) // F.m(Date) overrode E.m(Date) FW
			reverser: adapt LDT->Date
			invoke local F.m(Date)
			if return had changed, adapt return back

===

Migration step 3: source migration F.m(Date) -> F.m(LDT)
  source:
source:
    D.m(Date) -> D.m(LDT)
    E.m(Date) -> E.m(LDT)
    F.m(Date) -> F.m(LDT)
  class file:
    D.m(LDT) {}
    D.m(Date) FW -> D.m(LDT)
    E.m(LDT) {}
    E.m(Date) FW -> E.m(LDT)
    F.m(LDT) {}
    F.m(Date) FW -> F.m(LDT)
Runtime reversers created: None: all overriders of forwarders: also override target of forwarder

invocation	**dynamic receiver**	**resolution NOT invoked**	**selection: actual execution**
invokevirtual D::m(Date)	D	D.m(Date)	D.m(Date) forwarder: adapt Date->LDT invoke local D.m(LDT) if return had changed, adapt
	E	“”	E.m(Date) forwarder: adapt Date -> LDT invoke local D.m(LDT) if return had changed, adapt
	F	“”	F.m(Date) forwarder: adapt Date -> LDT invoke local F.m(LDT) if return had changed, adapt
invokevirtual E.m(Date)	E	E.m(Date)	E.m(Date) forwarder: adapt Date -> LDT invoke local m(LDT) if return had changed, adapt
	F	“”	F.m(Date) forwarder: adapt Date -> LDT invoke local F.m(LDT) if return had changed, adapt
invokevirtual F.m(Date)	F	F.m(Date)	F.m(Date) forwarder: adapt Date -> LDT invoke local F.m(LDT) if return had changed, adapt
invokevirtual D::m(LDT)	D	D.m(LDT)	D.m(LDT)
	E	“”	E.m(LDT)
	F	“”	F.m(LDT)

invocation	dynamic receiver	resolution NOT invoked	selection: actual execution
invokevirtual E.m(LDT)	E	E.m(LDT)	E.m(LDT)
	F	“”	F.m(LDT)
invokevirtual F.m(LDT)	F	F.m(LDT)	F.m(LDT)

By this time: all old clients go through forwarders. All the new clients go directly, which is the ultimate goal. Once all subtypes have migrated, we no longer need reversers. Once all clients have migrated we no longer need forwarders.

Point 1: The resolved method is NOT invoked, it is only the selected method that is invoked. We do NOT follow forwarding for the resolved method.
If the resolved method happens to also be the selected method, we will now execute it and will follow the forwarding.

Note, the same applies to fields - we will not get/set the resolved field. We will get/set the selected field, and follow the forwarding at that point.

Point 2: Hotspot’s vtable implementation is set up so that for
class E - a vtable (or itable) is a selection cache. It allows for fast virtual dispatch.
  For Hotspot, for class E, the vtable starts with the inherited vtable from superclass D.
  Any entries in the table are replaced when a method overrides an inherited method.
  Additional methods are appended.

So resolution gives you the offset in the vtable. Selection tells you which vtable owner to index based on that offset.

We KEEP the existing methods in the subclass so that they are executed exactly the same with no change in behavior (no exceptions due to narrowing etc.)

Agree that so far the only reverser need I have identified is when a class overrides a forwarder.

==========
Example II. Top-level super does two-step migration

original:
 F <: E <: D
 D.m(Date, Time)
 E.m(Date, Time)
 F.m(Date, Time)

Migration step 1: D.m(Date, Time) -> D.m(LDT, Time) // same as example I above, so skip this step
Migration step 2: E.m(Date, Time) -> E.m(Date, LDT)
class file:
   D.m(LDT, Time)

D.m(Date, Time) FW-> D.m(LDT, Time)
E.m(Date, LDT)
E.m(Date, Time) FW -> E.m(Date, LDT)
F.m(Date, Time)

Runtime reversers created:
E.m(LDT, Time) -> E.m(Date, Time) // E.m(Date, Time)FW overrides D.m(Date, Time)FW and does NOT have a local override for target D.m(LDT, Time) : create reverser
F.m(LDT, Time) -> F.m(Date, Time) // F.m(Date, Time) @ overrides D.m(Date, Time)FW and does NOT have a local override for target D.m(LDT, Time) : create reverser
F.m(Date, LDT) -> F.m(Date, Time) // F.m(Date, Time) overrides E.m(Date, LDT)FW and does NOT have a local override for target E.m(Date, LDT): create reverser

invocation	**dynamic receiver**	**resolution NOT invoked**	**selection: actual execution**
invokevirtual D::m(Date, Time)	D	D.m(Date,Time)	D.m(Date, Time) forwarder: adapt Date->LDT invoke local D.m(LDT, Time) if return had changed, adapt
	E	“”	E.m(Date, Time) forwarder: adapt Time -> LDT invoke local E.m(Date, LDT)) if return had changed, adapt
	F	“”	F.m(Date, Time) //original - unchanged behavior
invokevirtual E.m(Date, Time)	E	E.m(Date, Time)	E.m(Date, Time) forwarder: adapt Time -> LDT invoke local E.m(Date, LDT) if return had changed, adapt
	F	“”	F.m(Date, Time) //original - unchanged behavior
invokevirtual F.m(Date, Time)	F	F.m(Date, Time)	F.m(Date, Time) //original - unchanged behavior
invokevirtual D::m(LDT, Time)	D	D.m(LDT, Time)	D.m(LDT, Time)
	E	“”	E.m(LDT, Time) reverser: adapt LDT -> Date invoke local E.m(Date, Time) forwarder: adapt Time -> LDT invoke local E.m(Date, LDT) return adapt #2 return adapt #1

invocation	dynamic receiver	resolution NOT invoked	selection: actual execution
	F	“”	F.m(LDT, Time) reverser: adapt LDT -> Date invoke local F.m(Date, Time) return adapt
invokevirtual E.m(LDT, Time)	E	E.m(LDT, Time) reverser	E.m(LDT, Time) reverser: adapt LDT -> Date invoke local E.m(Date, Time) forwarder: adapt Time -> LDT invoke local E.m(Date, LDT) return adapt #2 return adapt #1
	F	“”	F.m(LDT, Time) reverser: adapt LDT -> Date invoke local F.m(Date, Time) return adapt
invokevirtual F.m(LDT, Time)	F	F.m(LDT, Time) reverser	F.m(LDT, Time) reverser: adapt LDT -> Date invoke local F.m(Date, Time) return adapt
invokevirtual E.m(Date, LDT)	E	E.m(Date, LDT)	E.m(Date, LDT)
	F	F.m(Date, LDT) reverser	F.m(Date, LDT) reverser: adapt LDT -> Time invoke local F.m(Date, Time) return adapt
invokevirtual F.m(Date, LDT)	F	F.m(Date, LDT) reverser	F.m(Date, LDT) reverser: adapt LDT -> Time invoke local F.m(Date, Time) return adapt

Migration step 1: D.m(Date, Time) -> D.m(LDT, Time) // same as example I above, so skip this step
Migration step 2: E.m(Date, Time) -> E.m(Date, LDT)
Migration step 3: D.m(LDT, Time) -> D.m(LDT, LDT)
  needs annotation: D.m(Date, Time) -> D.m(LDT, Time) -> D.m(LDT, LDT)
class file:
  D.m(LDT, LDT)
  D.m(LDT, Time) FW -> D.m(LDT, LDT)
// option A: keep intermediate forwarders
  D.m(Date, Time) FW -> D.m(LDT, Time)

  E.m(Date, LDT)
  E.m(Date, Time) FW -> E.m(Date, LDT)

  F.m(Date, Time)

Runtime reversers created:
Option A: keep intermediate forwarders
E.m(LDT, Time) -> E.m(Date, Time) // E.m(Date, Time) FW overrides D.m(Date, Time)FW -> D., (LDT, Time) AND there is no local override for D.m(Date, Time) other than a local forwarder which does not form a loop

NOW: E.m(LDT, Time) -> Em(Date, Time) reverser overrides D.m(LDT, Time)FW AND there is no local override for D.m(LDT, LDT)
E.m(LDT, LDT) -> E.m(LDT, Time) // create reverser (this does not form a local loop)

(Note: the source file has NO history of migration annotations that would tell me that we could go straight from E.m(Date, LDT) to E.m(LDT, LDT) and just adapt the first argument, so we have to go through multiple adaptor steps)

F.m(Date, LDT) -> F.m(Date, Time) //F.m(Date, Time) overrides E.m(Date,Time)FW -> F.m(Date, LDT)
F.m(LDT, Time) -> F.m(Date, Time) // F.m(Date, Time) @ overrides D.m(Date, Time) FW -> D.m(LDT, Time)

NOW: F.m(LDT, Time) -> F.m(Date, Time) reverser @ overrides D.m(LDT, Time)FW-> D.m(LDT, LDT) AND there is no local override for D.m(LDT, LDT)
F.m(LDT, LDT) -> F.m(LDT, Time) // create reverser (this does not form a local loop)

NOTE: Option B: lose intermediate forwarders and set up D.m(Date, Time) -> D.m(LDT, LDT) directly instead.
I do not have a way to get back the ability to make the intermediate calls with E and F receivers.

invocation	**dynamic receiver**	**resolution NOT invoked**	**selection: actual execution**
invokevirtual D::m(Date, Time)	D	D.m(Date,Time)	D.m(Date, Time) forwarder: adapt Date->LDT invoke local D.m(LDT, Time) forwarder: adapt Time->LDT invoke local D.m(LDT, LDT) return adapt #2 return adapt #1
	E	“”	E.m(Date, Time) forwarder: adapt Time -> LDT invoke local E.m(Date, LDT)) if return had changed, adapt
	F	“”	F.m(Date, Time) //original - unchanged behavior

invocation	dynamic receiver	resolution NOT invoked	selection: actual execution
invokevirtual E.m(Date, Time)	E	E.m(Date, Time)	E.m(Date, Time) forwarder: adapt Time -> LDT invoke local E.m(Date, LDT) if return had changed, adapt
	F	“”	F.m(Date, Time) //original - unchanged behavior
invokevirtual F.m(Date, Time)	F	F.m(Date, Time)	F.m(Date, Time) //original - unchanged behavior
invokevirtual D::m(LDT, Time)	D	D.m(LDT, Time)	D.m(LDT, Time) forwarder: adapt Time -> LDT invoke local D.m(LDT, LDT) return adapt
	E	“”	E.m(LDT, Time) reverser: adapt LDT -> Date invoke local E.m(Date, Time) forwarder: adapt Time -> LDT invoke local E.m(Date, LDT) return adapt #2 return adapt #1
	F	“”	F.m(LDT, Time) reverser: adapt LDT -> Date invoke local F.m(Date, Time) return adapt
invokevirtual E.m(LDT, Time)	E	E.m(LDT, Time) reverser	E.m(LDT, Time) reverser: adapt LDT -> Date invoke local E.m(Date, Time) forwarder: adapt Time -> LDT invoke local E.m(Date, LDT) return adapt #2 return adapt #1
	F	“”	F.m(LDT, Time) reverser: adapt LDT -> Date invoke local F.m(Date, Time) return adapt
invokevirtual F.m(LDT, Time)	F	F.m(LDT, Time) reverser	F.m(LDT, Time) reverser: adapt LDT -> Date invoke local F.m(Date, Time) return adapt
invokevirtual E.m(Date, LDT)	E	E.m(Date, LDT)	E.m(Date, LDT)

invocation	dynamic receiver	resolution NOT invoked	selection: actual execution
	F	F.m(Date, LDT) reverser	F.m(Date, LDT) reverser: adapt LDT -> Time invoke local F.m(Date, Time) return adapt
invokevirtual F.m(Date, LDT)	F	F.m(Date, LDT) reverser	F.m(Date, LDT) reverser: adapt LDT -> Time invoke local F.m(Date, Time) return adapt
invokevirtual D.m(LDT. LDT)	D	D.m(LDT, LDT)	D.m(LDT, LDT)
	E	E.m(LDT, LDT) reverser	E.m(LDT, LDT) reverser: adapt LDT -> Time invoke local E.m(LDT, Time) reverser: adapt LDT -> Date invoke local E.m(Date, Time) forwarder: adapt Time -> LDT invoke local E.m(Date, LDT) return adapt return adapt return adapt return adapt
	F	F.m(LDT, LDT)	F.m(LDT, LDT) reverser: adapt LDT -> Time invoke local F.m(LDT, Time) reverser: adapt LDT -> Date invoke local F.m(Date, Time) return adapt return adapt
invokevirtual E.m(LDT, LDT)	E	E.m(LDT, LDT)	E.m(LDT, LDT) reverser: adapt LDT -> Time invoke local E.m(LDT, Time) reverser: adapt LDT -> Date invoke local E.m(Date, Time) forwarder: adapt Time -> LDT invoke local E.m(Date, LDT) return adapt return adapt return adapt return adapt

invocation	dynamic receiver	resolution NOT invoked	selection: actual execution
	F	F.m(LDT,LDT)	F.m(LDT, LDT) reverser: adapt LDT -> Time invoke local F.m(LDT, Time) reverser: adapt LDT -> Date invoke local F.m(Date, Time) return adapt return adapt
invokevirtual F.m(LDT, LDT)	F	F.m(LDT, LDT)	F.m(LDT, LDT) reverser: adapt LDT -> Time invoke local F.m(LDT, Time) reverser: adapt LDT -> Date invoke local F.m(Date, Time) return adapt return adapt