Benchmarking current vs alternative version of Boogie

This compares the current version of Boogie with a an alternative, newer one.

In the current instance, we are comparing Boogie 2.8.31 with the head version in which *zero weights are used for array axioms*. The comparison uses the new monomorphized Boogie backend. There is a ~10% improvement visible from the benchmarks. While some verification problems take longer, overall verification time is reduced.

Preparation

Load the prover-lab crate. This may take *long* (minutes) the first time the Jupyter server is started because it compiles a lot Rust sources.

In [23]:
```
:sccache 1
:dep prover-lab = { path = "../.." }
```

Out[23]:
```
sccache: true
```

Make functions from the benchmark module available:

In [26]:
```
use prover_lab::benchmark::*;
```

Module Verification Time

In overall verification time for all Diem modules, zero-weight does about 10% better:

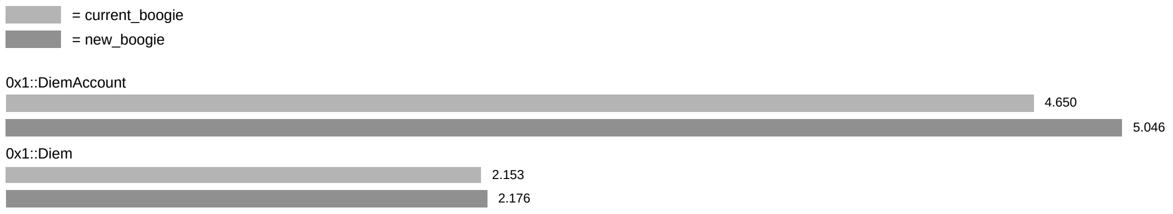
In [27]:
```
let mut current_mod = read_benchmark("current_boogie.mod_data")?;
let mut new_mod = read_benchmark("new_boogie.mod_data")?;
stats_benchmarks(&[&current_mod, &new_mod])
```

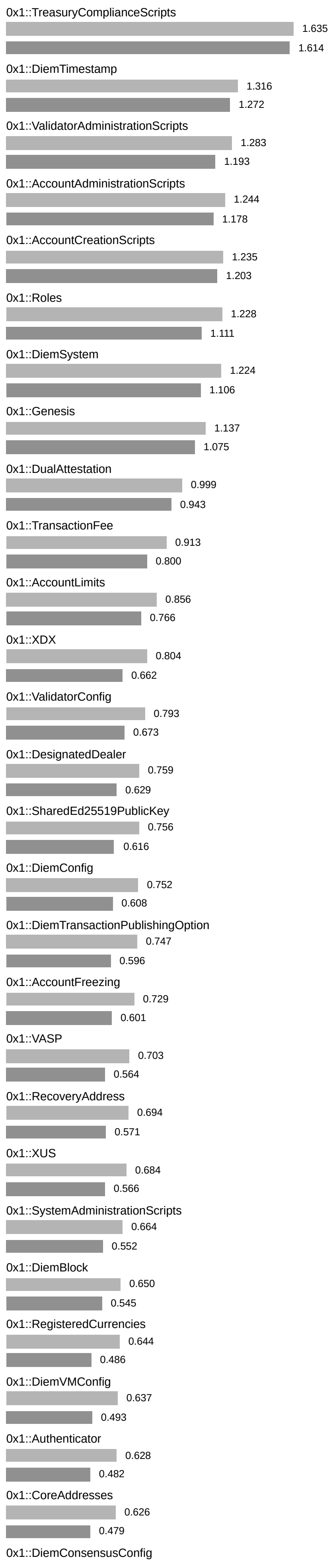
Out[27]:
```
current_boogie: 34.905s tot, 1.000 rel
new_boogie    : 31.556s tot, 0.904 rel
```

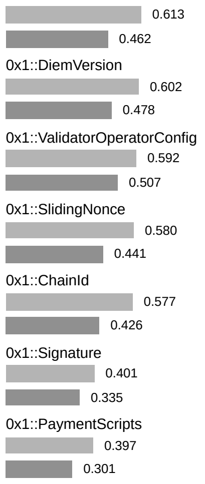
However, per module it appears that some of the more notrious difficult problems like DiemAccount takes longer. The advantage seems to be in speeding up simpler problems:

In [22]:
```
current_mod.sort(); // Will also determine order of other samples.
plot_benchmarks(&[&current_mod, &new_mod])
```

Out[22]:



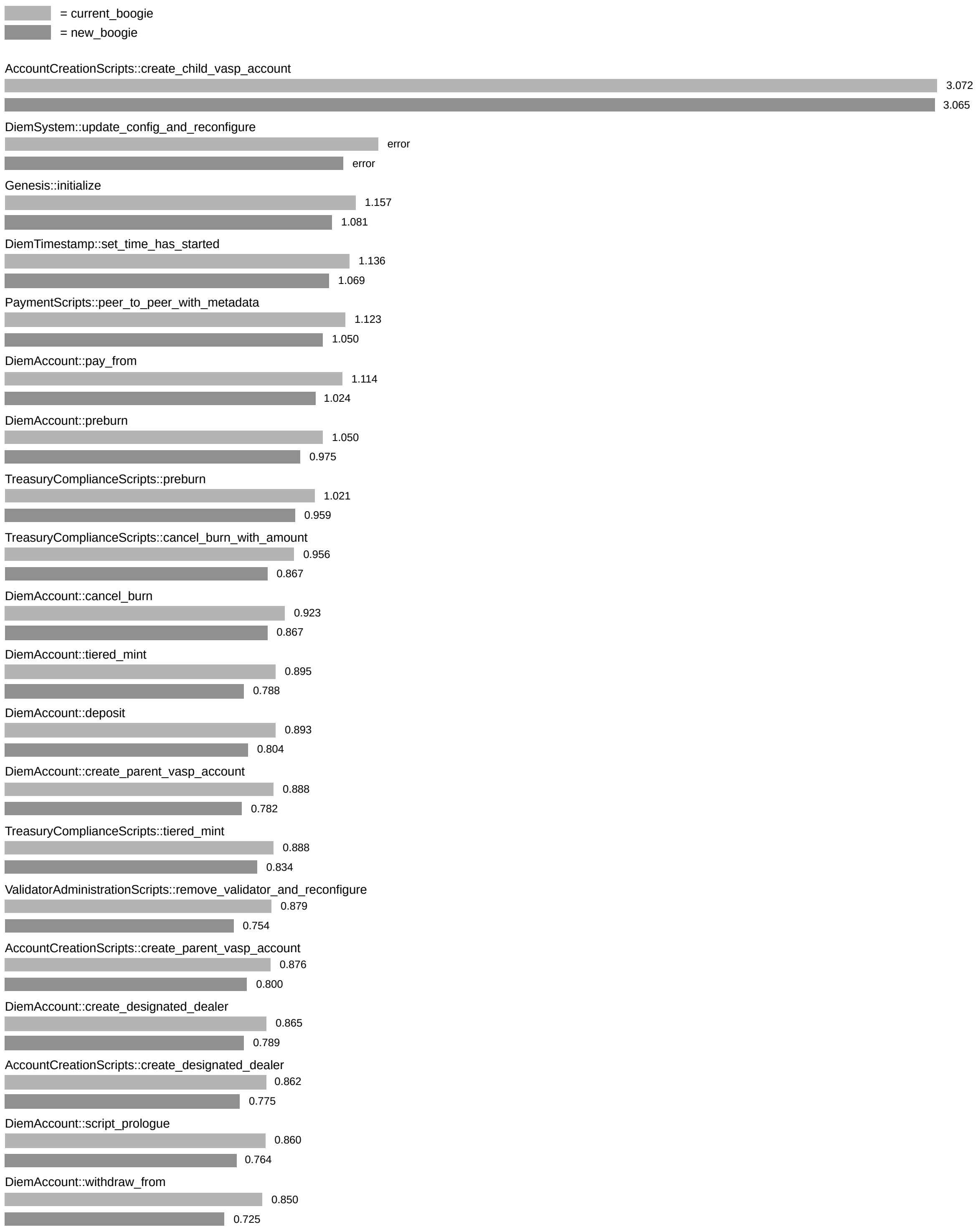

0x1::TreasuryComplianceScripts
1.635
1.614
0x1::DiemTimestamp
1.316
1.272
0x1::ValidatorAdministrationScripts
1.283
1.193
0x1::AccountAdministrationScripts
1.244
1.178
0x1::AccountCreationScripts
1.235
1.203
0x1::Roles
1.228
1.111
0x1::DiemSystem
1.224
1.106
0x1::Genesis
1.137
1.075
0x1::DualAttestation
0.999
0.943
0x1::TransactionFee
0.913
0.800
0x1::AccountLimits
0.856
0.766
0x1::XDX
0.804
0.662
0x1::ValidatorConfig
0.793
0.673
0x1::DesignatedDealer
0.759
0.629
0x1::SharedEd25519PublicKey
0.756
0.616
0x1::DiemConfig
0.752
0.608
0x1::DiemTransactionPublishingOption
0.747
0.596
0x1::AccountFreezing
0.729
0.601
0x1::VASP
0.703
0.564
0x1::RecoveryAddress
0.694
0.571
0x1::XUS
0.684
0.566
0x1::SystemAdministrationScripts
0.664
0.552
0x1::DiemBlock
0.650
0.545
0x1::RegisteredCurrencies
0.644
0.486
0x1::DiemVMConfig
0.637
0.493
0x1::Authenticator
0.628
0.482
0x1::CoreAddresses
0.626
0.479
0x1::DiemConsensusConfig



Top 20 by Function

In [7]:
```
let mut current_fun = read_benchmark("current_boogie.fun_data")?;
let mut new_fun = read_benchmark("new_boogie.fun_data")?;
current_fun.sort(); // Will also determine order of other samples.
current_fun.take(20);
plot_benchmarks(&[&current_fun, &new_fun])
```

Out[7]:


= current_boogie
= new_boogie
AccountCreationScripts::create_child_vasp_account
3.072
3.065
DiemSystem::update_config_and_reconfigure
error
error
Genesis::initialize
1.157
1.081
DiemTimestamp::set_time_has_started
1.136
1.069
PaymentScripts::peer_to_peer_with_metadata
1.123
1.050
DiemAccount::pay_from
1.114
1.024
DiemAccount::preburn
1.050
0.975
TreasuryComplianceScripts::preburn
1.021
0.959
TreasuryComplianceScripts::cancel_burn_with_amount
0.956
0.867
DiemAccount::cancel_burn
0.923
0.867
DiemAccount::tiered_mint
0.895
0.788
DiemAccount::deposit
0.893
0.804
DiemAccount::create_parent_vasp_account
0.888
0.782
TreasuryComplianceScripts::tiered_mint
0.888
0.834
ValidatorAdministrationScripts::remove_validator_and_reconfigure
0.879
0.754
AccountCreationScripts::create_parent_vasp_account
0.876
0.800
DiemAccount::create_designated_dealer
0.865
0.789
AccountCreationScripts::create_designated_dealer
0.862
0.775
DiemAccount::script_prologue
0.860
0.764
DiemAccount::withdraw_from
0.850
0.725

In [ ]: